МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет)_Автоматизації і комп'ютерних систем_____
Кафедра _Інформаційних технологій, штучного інтелекту і кібербезпеки_

«До захисту в ЕК»	**«До захисту допущено»**
Директор інституту(декан факультету)	Завідувач кафедри
_____ ___Андрій ФОРСЮК__	_____ _Сергій ГРИБКОВ__
(підпис) (ім'я та прізвище)	(підпис) (ім'я та прізвище)
«__» __лютого_____ 2024р.	«__» ____лютого_____ 2024р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

зі спеціальності__122 Комп'ютерні науки_____
(код та назва спеціальності)

освітньо-професійної програми_____Комп'ютерні науки_____

на тему: «Система інформаційної підтримки приймання та складування товарів у магазинах мережі АТБ»

Виконав: здобувач__3__ курсу, групи_5-ІТ_____

__Юрчик Володимир Володимирович_____
(прізвище, ім'я, по батькові повністю) (підпис)

Керівник __к.т.н.,доц. М'якшило Олена Михайлівна_____ ____
(прізвище , ім'я та по батькові повністю) (підпис)

Консультанти_____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

Рецензент _____ _____
(ім'я та прізвище) (підпис)

Я, як здобувач Національного університету харчових технологій розумію і підтримую політику університету з академічної доброчесності. Я не надавав і не одержував недозволеної допомоги під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Здобувач_____
(підпис)

Київ - 2024р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Автоматизації і комп'ютерних систем
Кафедра Інформаційних технологій, штучного інтелекту і кібербезпеки
Освітній ступінь бакалавр
Спеціальність 122 Комп'ютерні науки
Освітньо-професійна програма Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інформаційних технологій, штучного інтелекту і кібербезпеки
Сергій ГРИБКОВ

" 20 "листопада 2023 року

З А В Д А Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА

Юрчику Володимиру Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи Система інформацыйної підтримки приймання та складування товарів у магазинах мережі АТБ
керівник роботи М'якшило Олена Михайлівна, доц., к.т.н.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від 14 листопада 2023 року №924-кс
2. Строк подання здобувачем роботи 20 січня 2024 р.
3. Вихідні дані до роботи
Статут підприємства. Відомості про надходження матеріалів.
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити) С*истемний аналіз магазину мережі «АТБ»; розрахунок очікуваного економічного ефекту від впровадження системи; технічне завдання; розроблення функціональної моделі;алгоритмізація та реалізація комплексу задач автоматизації; обгрунтування вибору засобів розробки опис заходів з охорони праці.*
5. Перелік графічного матеріалу

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	к.т.н., доц. М'якшило О.М.		
2	к.т.н., доц. М'якшило О.М.		
3	к.т.н., доц. М'якшило О.М.		

7. Дата видачі завдання_____20 листопада 2023 року_____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Системний аналіз діяльності магазину мережі та постановка задачі на проектування	27.11.2023	Виконано
2	Розроблення функціональної моделі та аналіз існуючих бізнес процесів	02.12.2023	Виконано
3	Обгрунтування доцільності проєктування і розробки	05.12.2023	Виконано
4	Розрахунок економічного ефекту від впровадження системи	07.12.2023	Виконано
5	Технічне завдання на проєктування	11.12.2023	Виконано
6	Інформаційне забезпечення системи	13.12.2023	Виконано
7	Технічне та системне забезпечення розробки	23.12.2023	Виконано
8	Оформлення пояснювальної записки	04.01.2024	Виконано
9	Оформлення презентації	26.01.2024	Виконано

Здобувач _____ ___Володимир ЮРЧИК_____
(підпис) (прізвище та ініціали)

Керівник роботи _____ ___Олена М'ЯКШИЛО_____
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Метою кваліфікаційної бакалаврської роботи є створення інформаційної системи для підтримки процесів приймання та складування товарів у магазинах мережі АТБ. Розробка цієї інформаційної системи є важливою задачею для підвищення ефективності та якості логістики в мережі АТБ.

Об’єктом дослідження є процеси приймання та складування товарів у магазинах мережі АТБ, які включають в себе приймання товарів, їх розміщення на складі, облік кількості та стану товарів, а також взаємодію з постачальниками.

Результатом кваліфікаційної роботи буде розроблена інформаційна система, яка спростить та автоматизує процеси приймання та складування товарів у магазинах мережі АТБ. Система буде включати модулі для реєстрації нових товарів, внесення змін у наявні товари, контролю за станом товарів на складі, співпрацювати з постачальниками та генерувати звіти про стан товарообігу.

Реєстрація та облік товарів дозволить зберігати детальну інформацію про кожен товар, включаючи характеристики, кількість, термін придатності та постачальника. Система також допоможе контролювати рух товарів в магазині, їх розміщення та розподіл на полиці.

Розроблена інформаційна система має на меті поліпшити ефективність та точність управління товарообігом у магазинах мережі АТБ та сприяти оптимізації логістичних процесів.

Кваліфікаційна робота має обсяг у 69 сторінки, 4 таблиці, 16 рисунків, 2 додатки та базується на 20 літературних джерелах.

Ключові слова: ІНФОРМАЦІЙНІ СИСТЕМИ, ЛОГІСТИКА, ПРИЙМАННЯ ТА СКЛАДУВАННЯ ТОВАРІВ, МАГАЗИНИ, МЕРЕЖА АТБ, ОБЛІК ТОВАРІВ.

SUMMARY

The aim of the qualifying bachelor thesis is to create an information system to support the processes of receiving and storing goods in stores of the ATB network. The development of this information system is an important task for improving the efficiency and quality of logistics in the ATB network.

The object of the study is the processes of receiving and storing goods in the stores of the ATB network, which include receiving goods, placing them in the warehouse, accounting for the quantity and condition of goods, as well as interaction with suppliers.

The result of the qualifying bachelor thesis will be the development of an information system that will simplify and automate the processes of receiving and storing goods in the stores of the ATB network. The system will include modules for registering new goods, making changes to existing goods, monitoring the status of goods in the warehouse, cooperating with suppliers and generating reports on the status of goods circulation.

Product registration and accounting will allow you to store detailed information about each product, including characteristics, quantity, expiration date and supplier. The system will also help control the movement of goods in the store, their placement and distribution on the shelf.

The purpose of the developed information system is to improve the efficiency and accuracy of merchandise management in ATB chain stores and to facilitate the optimization of logistics processes.

The qualifying bachelor thesis has a volume of 69 pages, 4 tables, 16 figures, 2 appendices and is based on 20 literary sources.

Keywords: INFORMATION SYSTEMS, LOGISTICS, RECEIVING AND STORAGE OF GOODS, STORES, ATB NETWORK, ACCOUNTING OF GOODS.

ЗМІСТ

ВСТУП

У сучасному світі, де швидкість та ефективність логістичних процесів відіграють ключову роль у забезпеченні конкурентоспроможності підприємств на ринку, важливість автоматизації та оптимізації цих процесів не може бути переоцінена. Мережа супермаркетів АТБ, яка є одним з лідерів роздрібної торгівлі в Україні, стикається з необхідністю постійного вдосконалення своїх логістичних операцій, щоб забезпечити високий рівень обслуговування клієнтів та ефективне управління запасами.

Ця кваліфікаційна бакалаврська робота присвячена розробці та впровадженню інформаційної системи управління запасами для мережі АТБ. Метою роботи є аналіз існуючих проблем та визначення шляхів їх вирішення за допомогою сучасних інформаційних технологій. В роботі розглядаються питання автоматизації процесів обліку вантажів, оптимізації роботи складів, поліпшення процесів приймання, зберігання та відвантаження товарів.

Кваліфікаційна робота складається з кількох частин, які охоплюють теоретичні аспекти логістики та управління запасами, аналіз поточної ситуації в мережі АТБ, розробку вимог до нової системи та її архітектуру, а також практичну реалізацію проекту. Особлива увага приділяється питанням безпеки праці та техніки безпеки при роботі з інформаційною системою.

Важливим аспектом роботи є оцінка економічного ефекту від впровадження системи, що включає аналіз витрат на розробку, впровадження та експлуатацію системи, а також розрахунок очікуваних вигод і терміну окупності проекту. Результати кваліфікаційної роботи мають на меті не тільки підвищення ефективності логістичних процесів в АТБ, але й забезпечення основи для подальших досліджень та розвитку в області автоматизації логістики в роздрібній торгівлі.

Кваліфікаційна робота базується на актуальних даних, аналітичних дослідженнях та сучасних методиках в області інформаційних систем та логістики, що робить її релевантною для реальних потреб бізнесу та вносить вагомий вклад у розвиток ефективних логістичних рішень для мережі АТБ.

РОЗДІЛ 1 СИСТЕМНИЙ АНАЛІЗ МАГАЗИНУ МЕРЕЖІ «АТБ»

1.1 Загальна характеристика магазину мережі «АТБ»

Мережа супермаркетів АТБ успішно розвивається на ринку України з 1993 року. Головний дистрибуційний центр мережі, розташований в Дніпропетровській області, у місті Дніпро, є ключовим елементом логістики компанії. Цей логістичний комплекс має загальну площу понад 100 тисяч квадратних метрів і включає в себе множину складських приміщень, оснащених за останнім словом техніки.

Комплекс обладнаний найсучаснішими системами безпеки та підтримує цілодобову охорону, забезпечуючи надійне зберігання та обробку товарів. В рамках логістичного центру АТБ пропонується широкий спектр послуг: від складського зберігання до підготовки та відправлення товарів у роздрібні точки по всій країні.

Завдяки стратегічно вигідному розташуванню поруч з основними транспортними артеріями, логістичний центр ефективно з'єднаний з усіма регіонами України, що дозволяє оптимізувати логістику та забезпечити швидку доставку продукції до магазинів мережі.

Основні напрямки діяльності логістичного центру АТБ включають забезпечення потреб мережі в складському зберіганні, транспортуванні товарів, а також управлінні ланцюгами постачань. Завдяки цьому мережа АТБ здатна гарантувати високу якість продукції та її наявність у магазинах, задовольняючи потреби мільйонів українців щодня.

1.2 Організаційна структура магазину мережі «АТБ»

Магазин мережі «АТБ» використовує типову структуру управління , яка складається з наступних підрозділів:

Рисунок 1.1 — Верхній рівень організаційної структури

Управління складом здійснюється керуючим магазину.

1.3 Аналіз поточного стану комп'ютеризації підприємства

Сучасний рівень інформатизації в мережі супермаркетів АТБ потребує додаткового вдосконалення та уваги. Аналіз діяльності мережі виявив, що в АТБ активно використовуються різноманітні інформаційні системи та програмне забезпечення, спрямовані на автоматизацію процесів та підвищення загальної продуктивності.

Серед ключових систем, що застосовуються:

- **SAP ERP** - комплексне рішення для управління основними аспектами діяльності, включаючи фінанси, логістику, та продажі.
- **WMS (Warehouse Management System)** - програма для ефективного управління складськими запасами та оптимізації руху товарів.
- **CRM (Customer Relationship Management)** - інструмент для управління відносинами з клієнтами, спрямований на збільшення продажів та покращення сервісу.

Однак, існують певні недоліки в нинішній системі інформатизації, які стосуються непокриття всіх інформаційних потреб підрозділів. Виявлено, що деякі процеси все ще здійснюються вручну або з використанням застарілих методів, що призводить до затримок та помилок.

Особливо актуальним є питання впровадження системи для детального обліку вантажів. Відсутність такої системи ускладнює роботу співробітників та знижує загальну ефективність процесів логістики.

Вирішення цієї проблеми через розробку та впровадження комплексної системи обліку вантажів є критично важливим для оптимізації роботи мережі АТБ. Також необхідно провести детальний аналіз потреб кожного підрозділу та оновити існуюче програмне забезпечення або впровадити нові технологічні рішення для забезпечення безперебійної та продуктивної роботи.

Висновок про нинішній стан інформатизації в АТБ вказує на необхідність змін та оновлень, щоб забезпечити високий рівень ефективності та продуктивності в усіх аспектах діяльності мережі.

1.4 Розроблення функціональної моделі та аналіз існуючих бізнес процесів

1.4.1. Функціональна модель

Для дослідження процесу приймання та виконання замовлення в мережі супермаркетів АТБ використовуємо платформу моделювання бізнес-процесів Bizagi, що дозволяє візуалізувати процеси у нотації BPMN. Межі проекту охоплюють весь шлях від моменту звернення клієнта до фінального виконання замовлення, з точки зору керівника логістичного відділу АТБ.

Модель допоможе відповісти на такі питання:

- Які кроки необхідно виконати для обробки замовлення?
- Хто відповідає за виконання кожного з цих кроків?
- Які дані та інформація необхідні для ефективного виконання процесу?
- Яким правилам та нормативам слідує компанія при виконанні замовлення?
- Де можливе впровадження автоматизації та які аспекти процесу потребують удосконалення?

Контекстна діаграма у нотації BPMN представлена на рисунку 1.2.

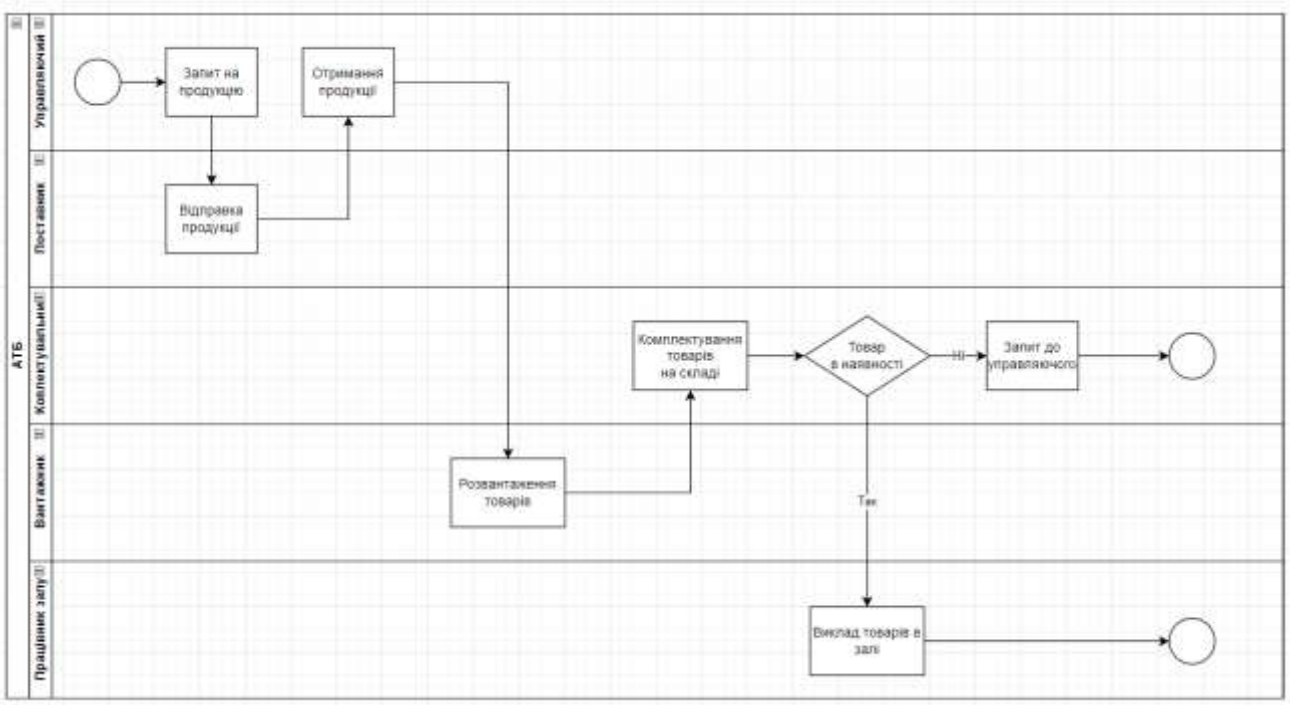

Рисунок 1.2 — Контекстна діаграма

1.4.2 Виявлені проблеми

На основі аналізу бізнес-процесів в мережі супермаркетів АТБ, особливо у відділі складського зберігання, було виявлено ряд проблем, які можуть негативно вплинути на ефективність логістики та загальну продуктивність мережі. Використані інформаційні системи для управління запасами та обліку вантажів мають такі недоліки:

1. Висока вартість - витрати на ліцензування та підтримку програмного забезпечення значно збільшують операційні витрати.
2. Відсутність безкоштовних версій (trial) - обмежує можливість оцінки функціоналу перед покупкою.
3. Складність у використанні - необхідність вкладення значних часових та фінансових ресурсів в навчання персоналу.
4. Незручний інтерфейс - знижує швидкість роботи та може призводити до помилок у введенні даних.

Системи, які застосовуються в АТБ, не завжди відповідають специфічним потребам логістичного комплексу через:

1. Невідповідність конкретним задачам - програмне забезпечення не покриває всіх специфічних потреб складського господарства АТБ.
2. Складність освоєння - системи вимагають тривалого часу на навчання персоналу, що знижує загальну продуктивність.
3. Незручний інтерфейс - ускладнює швидке використання функціоналу системи.
4. Обмежений функціонал - наявні системи не повністю задовольняють потреби в автоматизації складських процесів.

Запропоновані заходи для вирішення проблем:

- Підвищення кваліфікації персоналу - організація регулярних тренінгів та семінарів з використання інформаційних систем.
- Оновлення та модернізація технічного оснащення - інвестування в сучасні сканери штрих-кодів та інші пристрої для ефективного обліку та управління запасами на складі.
- Впровадження адаптованого або розробка спеціалізованого програмного забезпечення - розробка або кастомізація існуючих рішень, щоб вони відповідали конкретним потребам мережі АТБ, забезпечуючи інтуїтивно зрозумілий інтерфейс та необхідний функціонал.

Ці заходи дозволять оптимізувати процеси обліку вантажів, знизити ймовірність помилок, підвищити ефективність роботи складського комплексу та загальну продуктивність мережі супермаркетів АТБ.

1.5. Огляд існуючих рішень для розв’язання виявлених проблем

Для вирішення виявлених проблем у логістичних процесах мережі супермаркетів АТБ, особливо пов'язаних із обліком вантажів, складським управлінням, і ефективністю роботи персоналу, розглянемо наступні існуючі рішення на ринку.

1. SAP ERP (SAP Extended Warehouse Management)

Огляд: SAP ERP є комплексним рішенням для управління ресурсами підприємства, яке включає модуль SAP EWM (Extended Warehouse Management) для складського господарства. Система дозволяє автоматизувати бізнес-процеси, оптимізувати ланцюги постачання, управління запасами, планування та виконання складських операцій.

Сильні сторони:

- Висока інтеграція з іншими модулями SAP для комплексного управління підприємством.
- Гнучкість у налаштуванні бізнес-процесів, адаптація під специфіку діяльності.
- Розширені можливості для аналізу даних та звітності.

Слабкі сторони:

- Висока вартість ліцензування та впровадження.
- Складність в навчанні та освоєнні персоналом.
- Потреба в значних IT-ресурсах для підтримки та налаштування системи.

2. Microsoft Dynamics 365 Supply Chain Management

Огляд: Microsoft Dynamics 365 SCM є частиною більшої екосистеми Dynamics 365. Система надає широкі можливості для управління ланцюгами постачань, включаючи складське управління, управління виробництвом, планування та прогнозування.

Сильні сторони:

- Інтеграція з іншими продуктами Microsoft для забезпечення єдиного інформаційного простору.

- Можливість використання хмарних технологій для гнучкого масштабування.
- Доступність інструментів штучного інтелекту для оптимізації ланцюгів постачань.

Слабкі сторони:

- Вартість ліцензування може бути високою для малого та середнього бізнесу.
- Може вимагатися додаткова адаптація під конкретні бізнес-процеси.
- Потребує кваліфікованих фахівців для впровадження та налаштування.

3. Oracle Warehouse Management Cloud

Огляд: Oracle WMS Cloud є хмарним рішенням для управління складами, яке дозволяє автоматизувати складські операції, оптимізувати роботу з запасами та покращити обслуговування клієнтів.

Сильні сторони:

- Гнучкість хмарного рішення з можливістю масштабування під потреби бізнесу.
- Покращена видимість запасів та ефективність складських операцій завдяки інтеграції даних.
- Модульна структура, що дозволяє вибрати необхідний функціонал.

Слабкі сторони:

- Може вимагатися час на налаштування та адаптацію до специфіки бізнесу.
- Вартість хмарних сервісів залежить від обсягу використання та обраного пакету функцій.
- Потребує стабільного інтернет-з'єднання для доступу до системи.

1.6. Обґрунтування доцільності проєктування й розроблення

Проєктування та розроблення інформаційної системи для мережі супермаркетів АТБ є вкрай важливим з огляду на виявлені проблеми у сфері обліку вантажів та ефективності складської логістики. Цей процес не лише вирішить існуючі недоліки, але й сприятиме підвищенню загальної продуктивності та конкурентоспроможності мережі. Нижче наведено кілька ключових аргументів, що обґрунтовують доцільність такого проекту:

Підвищення ефективності внутрішніх процесів

Наявність інтегрованої системи дозволить автоматизувати численні ручні процеси, пов'язані з обліком та управлінням складом, що істотно підвищить ефективність роботи персоналу та знизить час на обробку замовлень.

Забезпечення точності даних

Автоматизація процесів допоможе забезпечити високу точність даних про запаси, відстеження вантажів та управління замовленнями, знизивши ризик помилок, що часто трапляються при ручному введенні інформації.

Зниження операційних витрат

Хоча первісні витрати на розробку та впровадження системи можуть бути значними, в довгостроковій перспективі це дозволить знизити загальні операційні витрати через оптимізацію складських процесів і зменшення витрат на персонал.

Покращення обслуговування клієнтів

Швидке та ефективне виконання замовлень, забезпечене завдяки автоматизованій системі, сприятиме покращенню задоволеності клієнтів та їх лояльності до мережі АТБ.

Адаптація до мінливих умов ринку

Гнучка та масштабована інформаційна система дозволить швидко адаптуватися до змін на ринку, враховуючи нові вимоги до логістики та управління запасами, а також впроваджувати інновації.

Обґрунтування інвестицій

Інвестиції в розробку індивідуальної інформаційної системи виправдані з огляду на специфіку та масштаби діяльності АТБ. Стандартні рішення часто не здатні повністю задовольнити унікальні потреби великої роздрібної мережі, тоді як спеціалізоване програмне забезпечення може бути точно налаштоване під ключові бізнес-процеси компанії.

Розроблення та впровадження спеціалізованої інформаційної системи для АТБ стане вирішальним кроком на шляху до оптимізації логістичних та складських процесів, забезпечення високого рівня задоволеності клієнтів та зміцнення позицій компанії на ринку.

1.7 Розрахунок економічного ефекту від впровадження системи

Визначення економічного ефекту від впровадження інформаційної системи для мережі АТБ є ключовим аспектом, що лежить в основі техніко-економічного обґрунтування розробки та впровадження автоматизованої системи управління складом. Для АТБ джерелами прибутку від впровадження такої системи можуть бути:

- Скорочення часу обробки замовлень: оптимізація логістичних процесів дозволить зменшити час від моменту отримання замовлення до його виконання.
- Збільшення обсягу оброблюваних замовлень: автоматизація дозволить одночасно обробляти більше замовлень без збільшення штату працівників.
- Зниження витрат на зберігання: ефективне управління складськими запасами знизить потребу в зайвих площах для зберігання.
- Оптимізація закупівель: можливість аналізу даних для пошуку більш вигідних умов закупівлі ресурсів.
- Раціоналізація процесів: скорочення кількості непотрібних процедур, робіт та операцій для зниження загальних витрат.

Ступінь новизни розроблюваної системи для АТБ можна класифікувати як "В" — адаптація та модифікація існуючих проектних рішень з урахуванням специфіки та потреб мережі. Це передбачає використання перевірених технологій та підходів, але з необхідними змінами для відповідності унікальним вимогам бізнесу.

Група складності алгоритму для розробки автоматизованої системи в АТБ визначається як група 1, що відображає високий рівень стандартизації та автоматизації процесів з використанням передових технологічних рішень.

Для системи обліку вантажів мережі супермаркетів АТБ, адаптуючи наведену інформацію під конкретні потреби та умови, можемо представити узагальнені дані вхідної та вихідної інформації як у таблиці нижче.

Таблиця. 1.1

Узагальнені дані для вхідної та вихідної інформації системи обліку вантажів мережі АТБ:

Вид інформації	Позначення	К-сть наборів даних
Змінна інформація	ЗІ	m=8
Нормативно – довідкова інформація	НДІ	n=3
Банк (база) даних	БД	p=1
Обробка в режимі реального часу	РЧ	Так
Забезпечення телекомунікаційної обробки даних і управління віддаленими об’єктами	ТОУ	Так

Таблиця 1.2

Визначення витрат часу для системи підтримки роботи логістичного центру АТБ:

Вид системи	Стадія розробки системи	
	Ескізний проект	Технічне завдання
Управління логістикою АТБ: управління транспортом, технічне обслуговування, управління перевезеннями; управління зберіганням	T1=80	T2=50

Вхідні дані для визначення витрат часу на стадіях «технічний проект», «робочий проект» і «впровадження»:

- кількість форм вхідної інформації: 8;
- кількість форм вихідної інформації: 2;

- базове значення витрат часу для стадії «Технічний проєкт» ТБ3=150
- базове значення витрат часу для стадії «Робочий проєкт» ТБ4=220
- базове значення витрат часу для стадії «Впровадження» ТБ5=65

– Визначення витрат часу для стадії "технічний проект" (Т3).

$$T_3 = T_{Б3} * k_п * k_о$$

$$k_п = \frac{k_1 * m + k_2 * n + k_3 * p}{m + n + p}$$

$$k_п = \frac{(1.0 + 0.72 * 2 + 2.08 * 1)}{(8 + 2 + 1)} = 11.52/10 = 1.48$$

$$T_3 = 150 * 1.048 * 1.26 = 198$$

– Визначення витрат часу на стадії «робочий проект» (Т4).

$$k_п = \frac{k_1 * m + k_2 * n + k_3 * p}{m + n + p}$$

Таблиця 1.3 Коефіцієнти k1(ЗІ), k2(НДІ), k3(БД) для стадії «робочий проєкт»

Вид використаної інформації	Ступінь новизни
	В
k_1(ЗІ)	1.2
k_2(НДІ)	0.65
k_3(БД)	0.54

$$k_п = \frac{(1.2 * 8 + 0.65 * 2 + 0.54 * 1)}{(8 + 2 + 1)} = 11.24/11 = 1.04$$

$$T_4 = T_{Б4} * k_п * k_о * k_с$$

Для знаходження $k_с$ для формули необхідно ідентифікувати складність контролю вхідної та вихідної інформації.

Тобто $k_с = 1.6$

$$T_4 = 206 * 1.04 * 1.26 * 1.6 = 462{,}336$$

– Визначення витрат часу на стадії «впровадження» (Т5).

$$k_п = \frac{k_1 * m + k_2 * n + k_3 * p}{m + n + p}$$

$$T_5 = T_{Б5} * k_п * k_о * k_с$$

$$T_5 = 65 * 0{,}989 * 1.21 * 1.6 = 127{,}3$$

Отже, загальні витрати людської праці складають визначаються:

$$Т_∑ = Т_1 + Т_2 + Т_3 + Т_4 + Т_5$$

$$Т_∑ = 70 + 43 + 180 + 445 + 117 = 855$$

Визначимо чисельність виконавців Ч.

$$Ч = \frac{Т_∑}{Ф}$$

Якщо для виконаня роботи припустимо кількість робочих годин складає 530 із 7-годинним робочим днем, тому на розробку проекту виділено Ф, днів:

Ф = 530/7 = 75 днів

Для дипломного проекту Ф = 75 днів. Тоді визначаемо кількість місяців із розрахунку 25 робочих днів.

Кількість місяців на розробку, М: ***М = Ф/25 = 75/25 = 3 місяці***

Отже, для виконання такого проекту потрібно таку чисельність виконавців Ч, виконавців, обраховується за формулою:

$$Ч = \frac{Т_∑}{Ф}$$

Ч = 914,6/75 = 12 днів

Приймемо 25000 грн за заробітну плату програміста, тоді загальна сума заробітних плат, які будуть видані програмістам складає:

$$\boldsymbol{V_1'} = Ч * М * ЗП = \mathbf{21 * 3 * 25000 = 900000}$$ ***грн***

Витрати, пов'язані з розробкою програми на ПК

Розрахунок річного фонду часу роботи ПК

Дійсний річний фонд часу ПК у годинах дорівнює числу робочих годин у році для оператора, за винятком часу на технічне обслуговування і ремонт ПК (в середньому 5год/міс + 6 роб.днів/рік).

$Т_{ПК}$ = 2000-(6*8+5*12) = ***1892*** год.

Оскільки під час виконання дипломного проекту (роботи) студент в середньому витрачає 450 год. машинного часу, то величина фонду часу ПК дорівнює

$Т'_{пк}$ = 1892*(450/2000) = 425.7 год

Поточні витрати на експлуатацію V "

Балансована вартість ПК, де Цр - ринкова вартість ПК, орієнтовно складає 40000 грн, $к_{ун}$ – коефіцієнт, що враховує витрати на установку ПК . кун=0,12

$Ц_{ПК} = Ц_Р * (1 + k_{УН})$ = 40000*(1+0,12) = 44800 грн

Амортизаційні відрахування використання ПК, $З_{АМ,}$ обчислюються за формулою $З_{АМ} = \frac{Ц_{ПК}}{H_А}$ =44800/5 = 8960 грн

Витрати на електроенергію ($З_{ЕЛ}$), споживану ПК, обчислюються

$$З_{ЕЛ} = Р_{ПК} * Т_{ПК} * Ц_{ЕЛ} * А$$

, де потужність ПК, $Р_{ПК}$ = 0.5 кВт; фонд корисного часу роботи ПК, $Т_{ПК}$ = 435.16 год, вартість 1 кВт електроенергії для підприємств, $Ц_{ЕЛ}$ = 1,86 грн/кВт, коефіцієнт інтенсивного використання ПК, А = 0.9.

$З_{ЕЛ} = 0{,}5 * 435.16 * 1.68 * 0.9$ = 329 грн

Витрати на поточний ремонт і технічне обслуговування ПК ($З_Р$)визначаються як 6% від балансової вартості ПК, $Ц_{ПК}$.

$$З_р = Ц_{ПК} * 0.06$$

$$З_р = 44800 * 0.06 = 2688 \text{ грн}$$

Непрямі витрати, пов’язані з експлуатацією ПК ($З_{МАТ)}$, визначаються як 5% від балансової вартості ПК $Ц_{ПК}$.

$$З_{МАТ} = Ц_{ПК} * 0.05$$

$$З_{МАТ} = 44800 * 0.05 = 2240 \text{ грн}$$

Поточні витрати на експлуатацію V "

$$V''_1 = З_{ОП} + З_{АМ} + З_{ЕЛ} + З_{Р} + З_{МАТ}$$

Заробітна плата обслуговуючого персоналу складає в середньому - 12000

Тож, поточні витрати на експлуатацію, V_1'', грн, складають:

$$V_1'' = 12000 + 8960 + 329 + 2688 + 2240 = 25\ 217 \text{ грн}$$

А, загальні витрати на розробку програмного забезпечення комп’ютерної системи складуть:

$$V_1 = V_1' + V_1'' = 900\ 000 + 25\ 217 = 925\ 217 \text{ грн}$$

Розрахунок витрат на придбання і установку ПК

V_2 = $Ц_{ПК}$ = 50000 грн

Якщо немає потреби в купівлі ПК, то ці витрати дорівнюють «0».

Розрахунок витрат на підготовку приміщення і навчання персоналу

Витрати на підготовку приміщення V_3:

V_3 = 0 так як приміщення є в наявності.

Витрати на навчання персоналу V_4

В середньому навчання персоналу триватиме 1 місяць, тому можна вважати, що V_3 = 5000 грн;

Загальна вартість розробки і впровадження системи вираховується за формуло:.

$$V_\Sigma = V_1 + V_2 + V_3 + V_4$$

$$V_\Sigma = 925\ 217 + 55\ 000 + 0 + 5000 = 985\ 217 \text{ грн}$$

Оскільки норма амортизаційних втрат для комп’ютерних систем НА = 5, то для обрахування річного економічного ефекту слід брати до розгляду величину:

$$V_р = \frac{V_\Sigma}{Н_А}$$

$$V_{\text{р}} = \frac{985\ 217}{5} = 197\ 043\ \text{грн}$$

Термін окупності розробки визначається:

$\text{Т}_{\text{ок}} = \frac{\mathbf{1}}{\text{К}_{\text{ЕФ}}}$, де коефіцієнт економічної ефективності $\text{К}_{\text{ЕФ}} = \frac{\text{П}_{\text{р}}}{V_{\text{р}}}$, де річний прибуток $\text{П}_{\text{р}}$ від впровадження системи буде досягнуто за рахунок збільшення кількості працюючих, як результат можливість виконувати більше замовлень і орієнтовно складатиме 90 000 грн на рік.

$$\text{К}_{\text{ЕФ}} = \frac{100\ 000}{180\ 743} = 0.507$$

Отже, термін окупності ІС буде складати:

$$\text{Т}_{\text{ок}} = \frac{\mathbf{1}}{\mathbf{0.507}} = \mathbf{1.97}\ \text{років}$$

РОЗДІЛ 2. ТЕХНІЧНЕ ЗАВДАННЯ НА ПРОЄКТУВАННЯ

1. Загальні положення.

1.1. Найменування системи: Інформаційна система управління запасами мережі АТБ. Це технічне завдання (далі – ТЗ) розроблено у відповідності з вимогами ДСТУ 34.602-89 і є основним документом, який визначає вимоги та порядок проектування, розробки, та впровадження інформаційної системи для управління запасами мережі супермаркетів АТБ.

1.2. Результати робіт зі створення системи оформлюються згідно з вимогами ДСТУ на відповідні етапи розробки. Порядок оформлення і передачі результатів у даному випадку визначається змістом і календарним планом виконання розробки.

1.3. У випадку необхідності, на наступних стадіях робіт по створенню системи, окремі положення можуть уточнюватися і розвиватися.

2. Призначення і цілі створення системи:

2.1. Призначення системи. Призначення інформаційної системи управління запасами мережі АТБ полягає у поліпшенні управління складською логістикою та ефективності загальної роботи мережі. Застосування сучасних інформаційних технологій дозволить автоматизувати ключові процеси управління запасами, від приймання товару до його розміщення на складі та відправлення в магазини, що сприятиме зниженню ризику помилок та покращенню оперативності реагування на зміни запасів.

2.2. Цілі створення системи. Основною метою створення системи є автоматизація та оптимізація процесів складського управління в мережі АТБ, включаючи ефективний прийом, зберігання та відправлення товарів. Це дозволить підвищити загальну продуктивність логістичної системи мережі, забезпечити точність управління запасами та підвищити рівень задоволеності клієнтів за рахунок покращення доступності товарів.

3. Характеристика об’єкта автоматизації.

3.1. Короткі відомості про об’єкт автоматизації. Об’єктом автоматизації є логістичні процеси мережі супермаркетів АТБ. Базовий об’єкт впровадження — ПАТ "ЛОГІСТИКА АТБ".

4. Вимоги до системи

4.1. Вимоги до системи в цілому. Система має надавати можливість додавання, оновлення, видалення записів по товарах, а також повинна бути інтегрована зі складом, транспортними процесами тощо. Робочі місця користувачів системи можуть розташовуватись у мережах загального користування.

4.1.1. Вимоги до структури і функціонування системи.

4.1.1.1. Система має базуватися на клієнт-серверній архітектурі, використовуючи єдину базу даних (БД). Вона має бути інтегрована з:

- Відділом складу;
- Відділом логістики;
- Відділом управління ланцюгом постачання.

4.1.1.2. Діагностика функціонування системи має включати виявлення відхилень від норми в процесах розв’язання задач, порушень у роботі комп’ютерно-технічних засобів та програмних помилок, надаючи користувачам відповідні діагностичні повідомлення.

4.1.1.3. Розвиток і модернізація системи мають здійснюватися на основі детального аналізу її поточного стану та визначення вимог і потреб щодо управління запасами. Це передбачає збір вимог від користувачів, ідентифікацію слабких місць, а також визначення цілей модернізації.

4.1.1.4. Система має забезпечувати діалогову та мережну (розподілену) обробку даних, інтеграцію з системами транспортування, засоби для аналітики та

звітності, щоб оптимізувати логістичні процеси та підвищити ефективність роботи логістичних центрів АТБ.

РОЗДІЛ 3
ОПИС КОМПЛЕКСУ ЗАДАЧ АВТОМАТИЗАЦІЇ

3.1. Інформаційне забезпечення системи

На рисунку 3.1 представлено схему БД

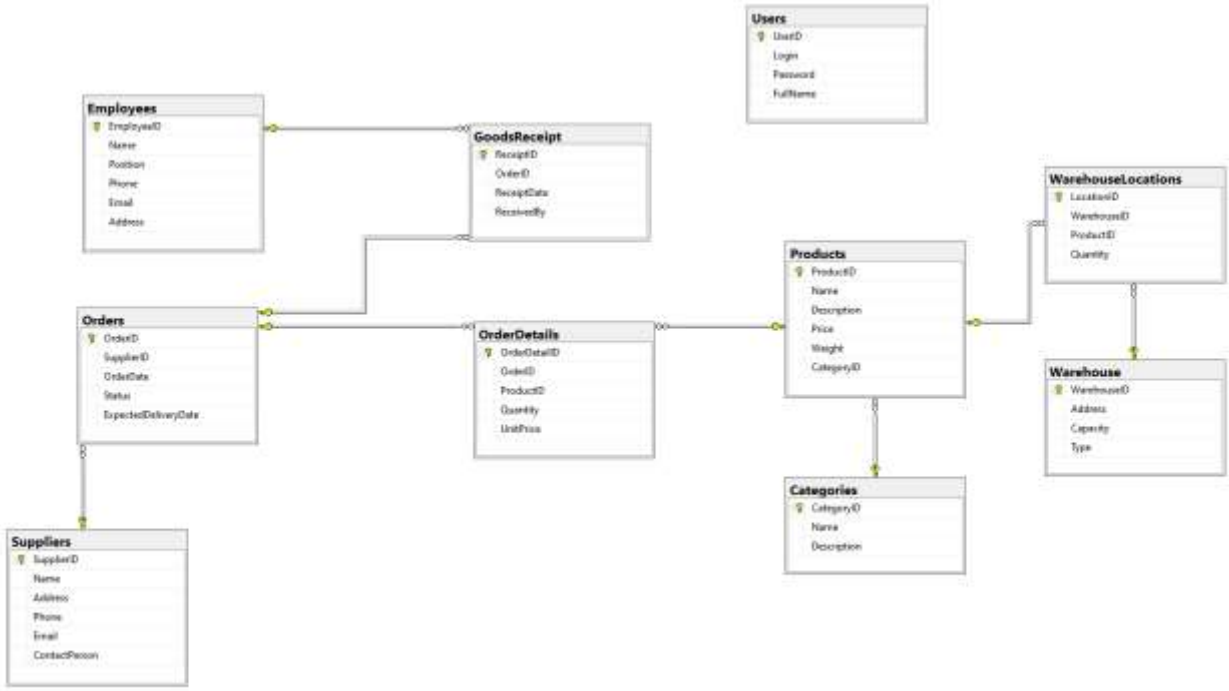

Рисунок 3.1 — Схема бази даних

Для роботи з БД слід під'єднати її до проекту в MS Visual Studio.

Після створення та завантаження бази даних на MS SQL Server 2019, ми підключаємо Visual Studio 2022 для подальшої розробки системи. Для підключення бази даних до середовища Microsoft Visual Studio 2022 використовується компонент "Data Source", де ми обираємо додати нове джерело та вводимо назву сервера та БД. Перевіряємо підключення та починаємо роботу.

Під'єднуємо створенні таблиці до DataSet Microsoft Visual Studio 2022.

DataSet в MS Visual Studio представлено на рисунку 3.2.

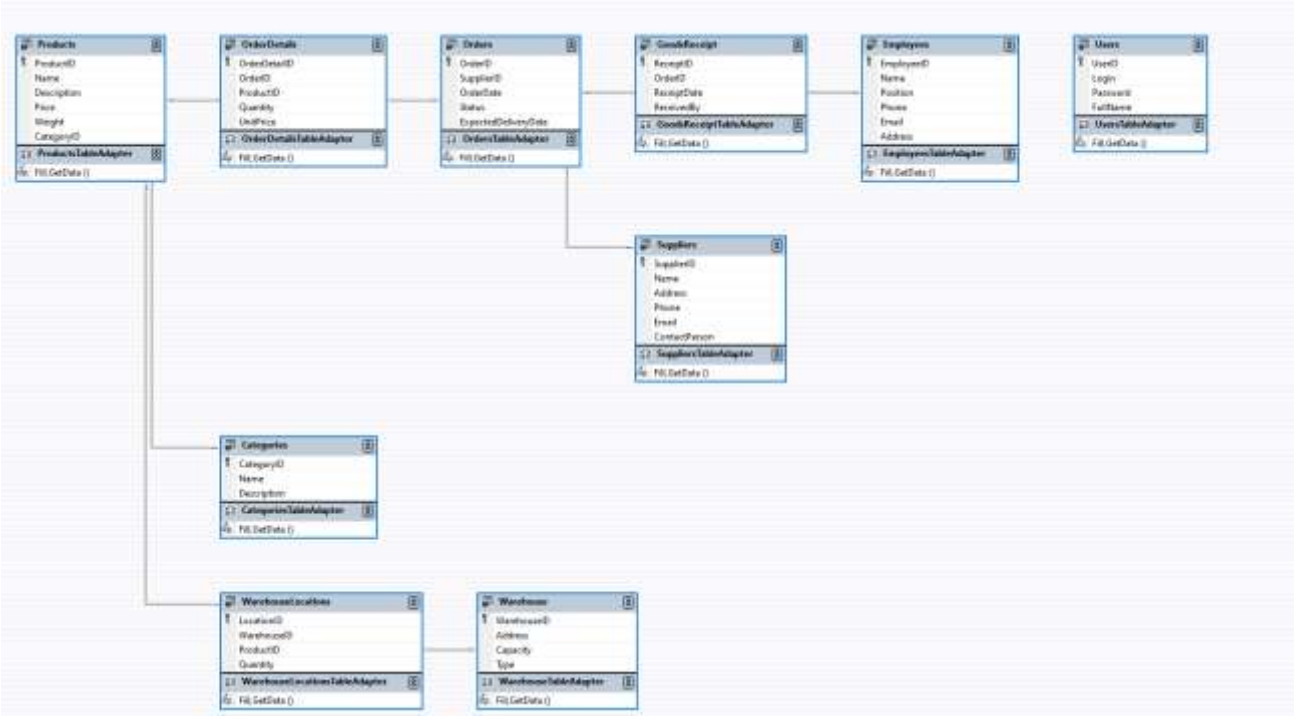

Рисунок 3.2 — DataSet в MS Visual Studio

3.2. Алгоритмізація та реалізація комплексу задач автоматизації

Система працює шляхом навігації за допомогою відповідних кнопок меню та переходу до різних вкладок, де відображаються різноманітні форми.

ІС має систему авторизації.

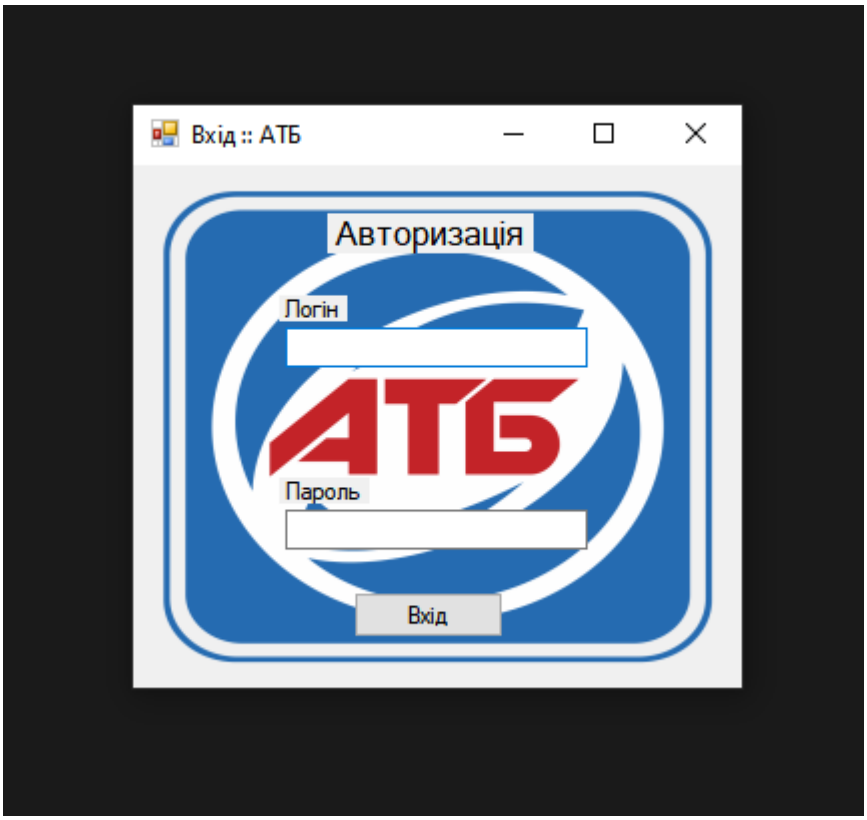

Рисунок 3.1 — Вікно авторизації

Код авторизації має наступний вигляд:

```
private void button1_Click(object sender, EventArgs e)
{
    string connectionString = "Data Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\\StoragingDB.mdf;Integrated Security=True";
    string query = "SELECT COUNT(1) FROM Users WHERE Login=@login AND Password=@password";
```

```
            using (SqlConnection conn = new SqlConnection(connectionString))
            {
                conn.Open();
                using (SqlCommand cmd = new SqlCommand(query, conn))
                {
                    cmd.Parameters.AddWithValue("@login", textBox1.Text.Trim());
                    cmd.Parameters.AddWithValue("@password", textBox2.Text.Trim());

                    int result = Convert.ToInt32(cmd.ExecuteScalar());
                    if (result == 1)
                    {
                        MessageBox.Show("Успішний вхід!");
                        this.Hide();
                        MainForm mainForm = new MainForm();
                        mainForm.ShowDialog();
                        this.Show();

                    }
                    else
                    {
                        MessageBox.Show("Логін або пароль невірний.");
                    }
                }
            }
        }
```

Після авторизації в системи користувач отримує доступ до роботи з базою даних.

Рисунок 3.2 — Успішна авторизація

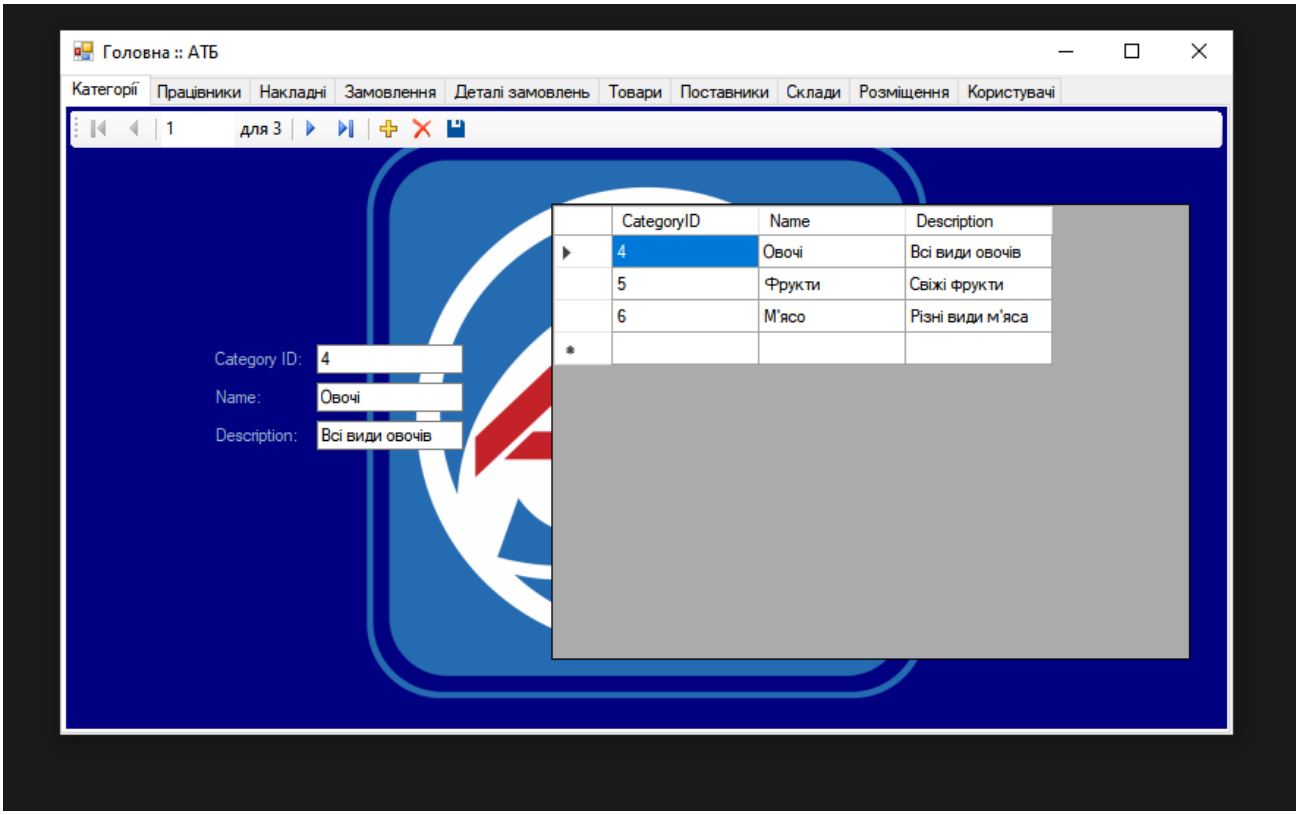

Рисунок 3.3 — Вікно категорій товарів

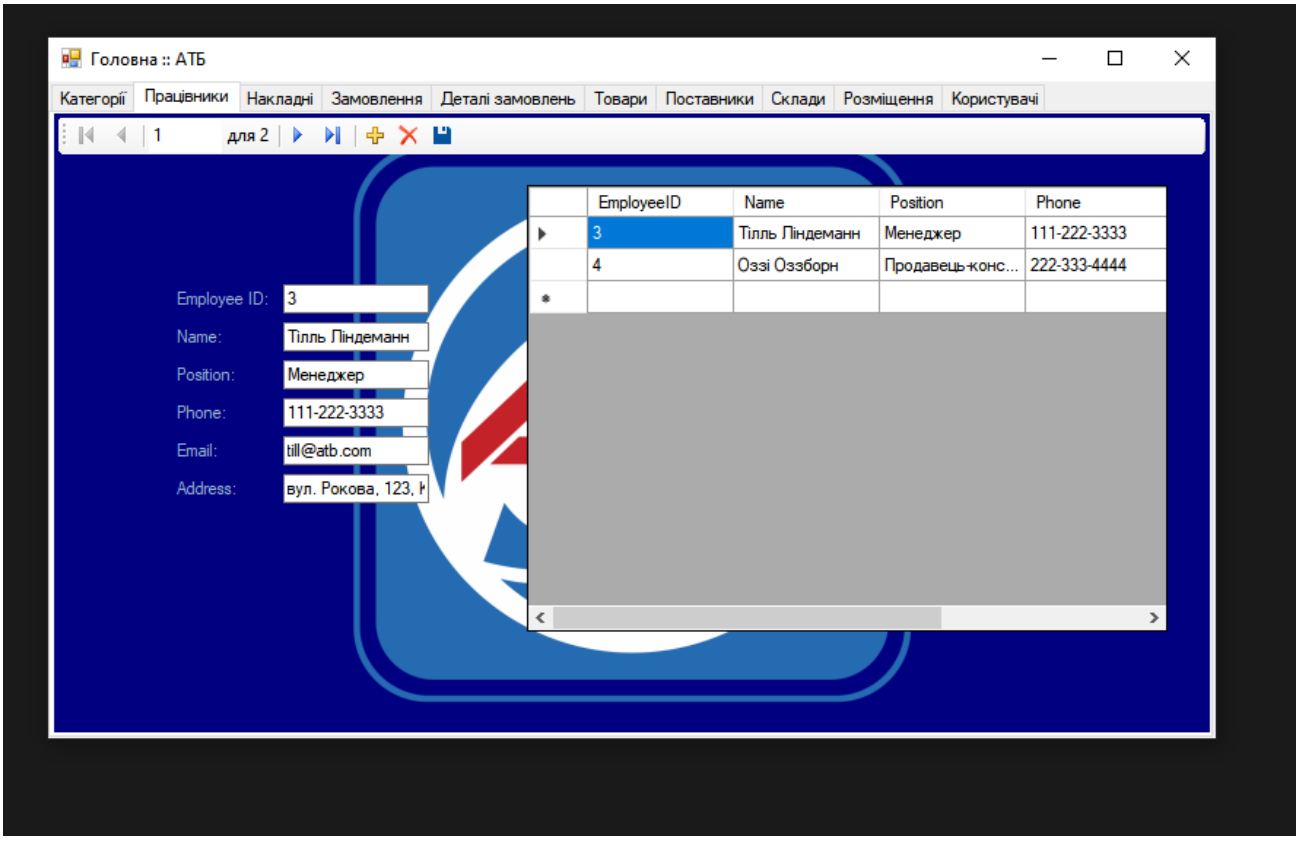

Рисунок 3.4 — Вінок працівників

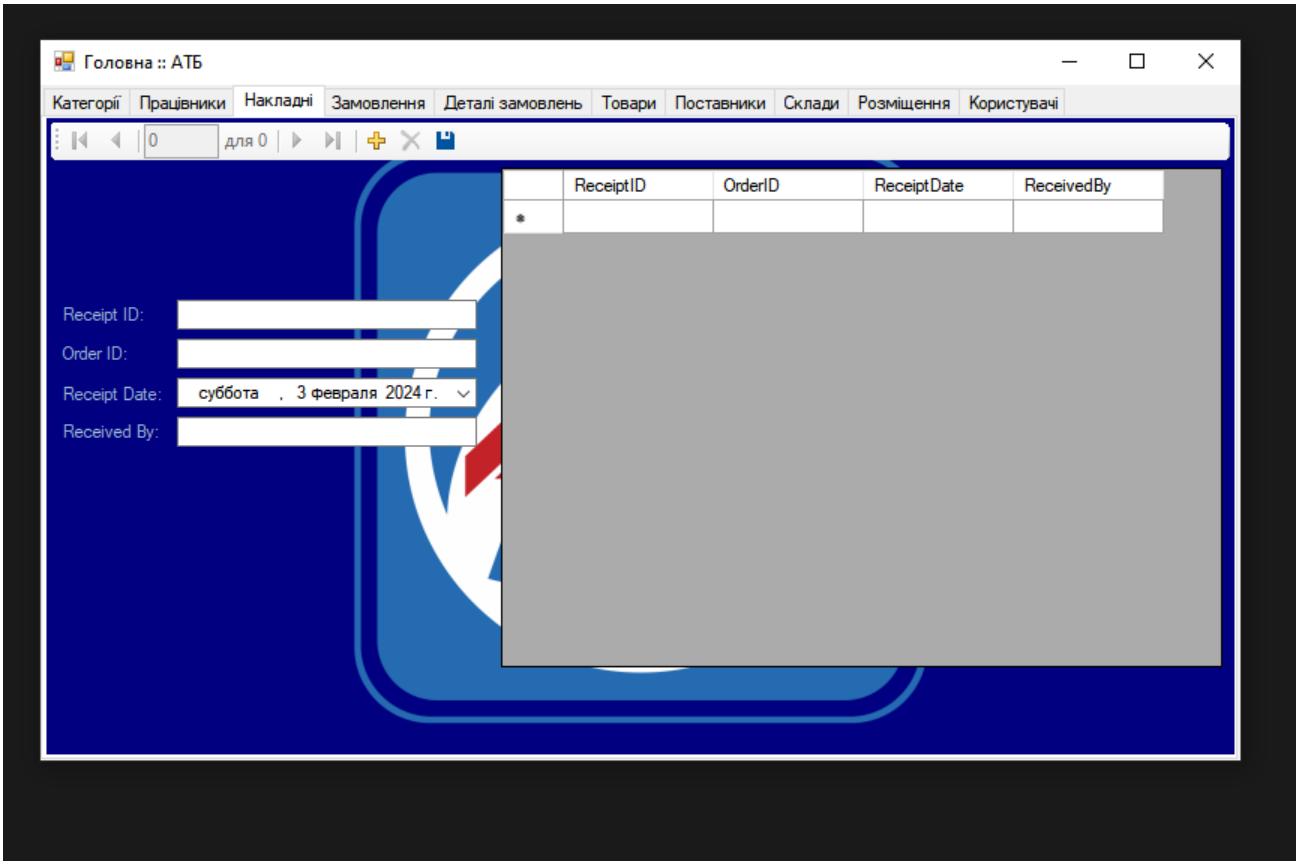

Рисунок 3.5 — Вікно накладних

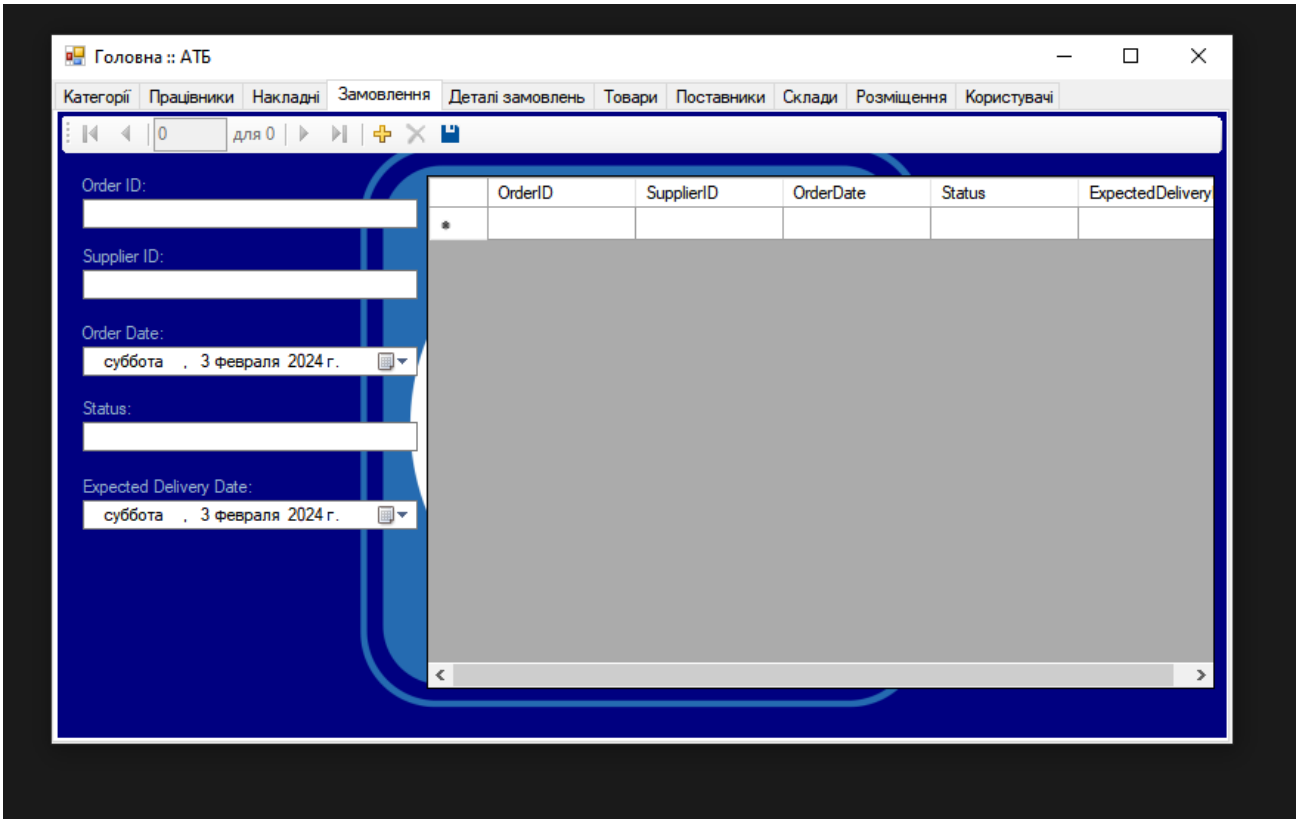

Рисунок 3.6 — Вінок замовлень

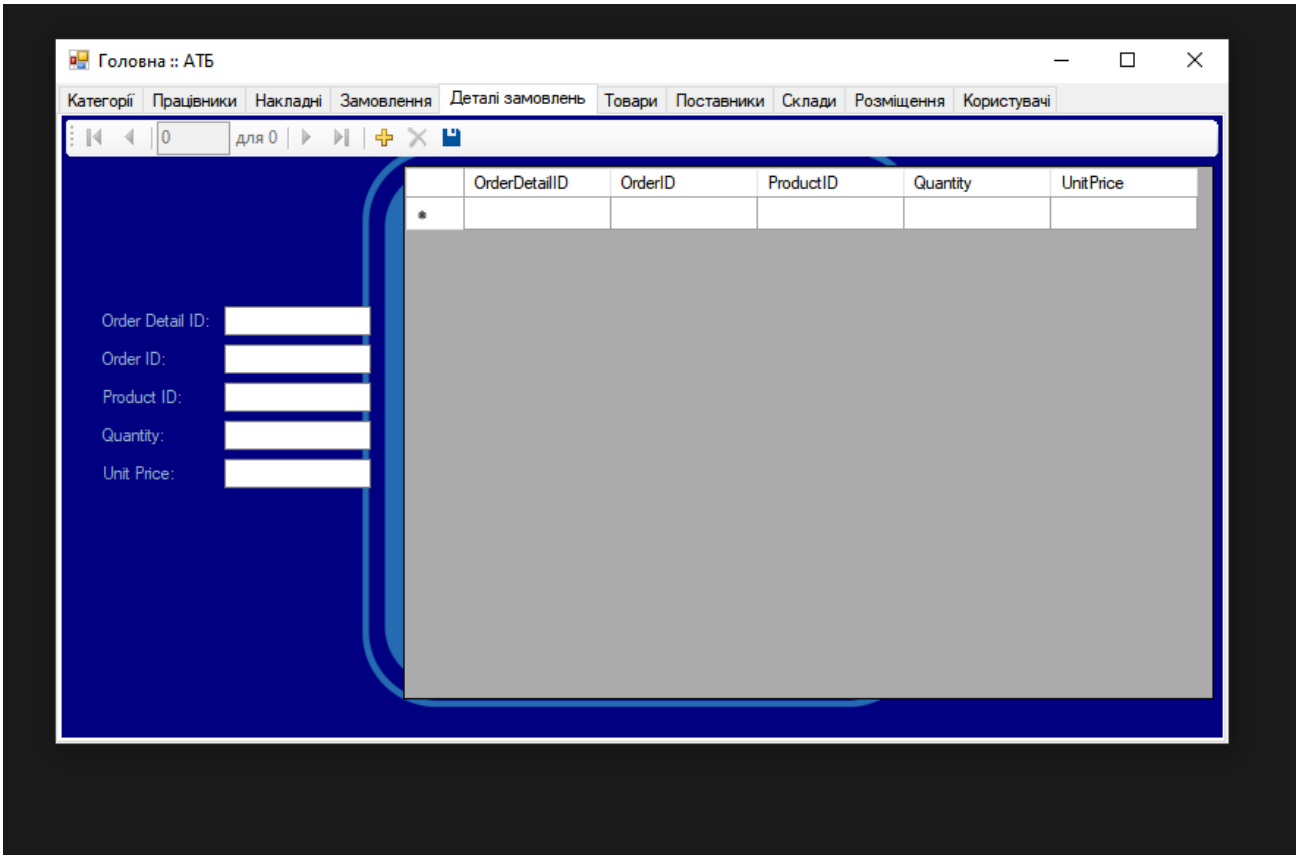

Рисунок 3.7 — Вікно деталей замовлень

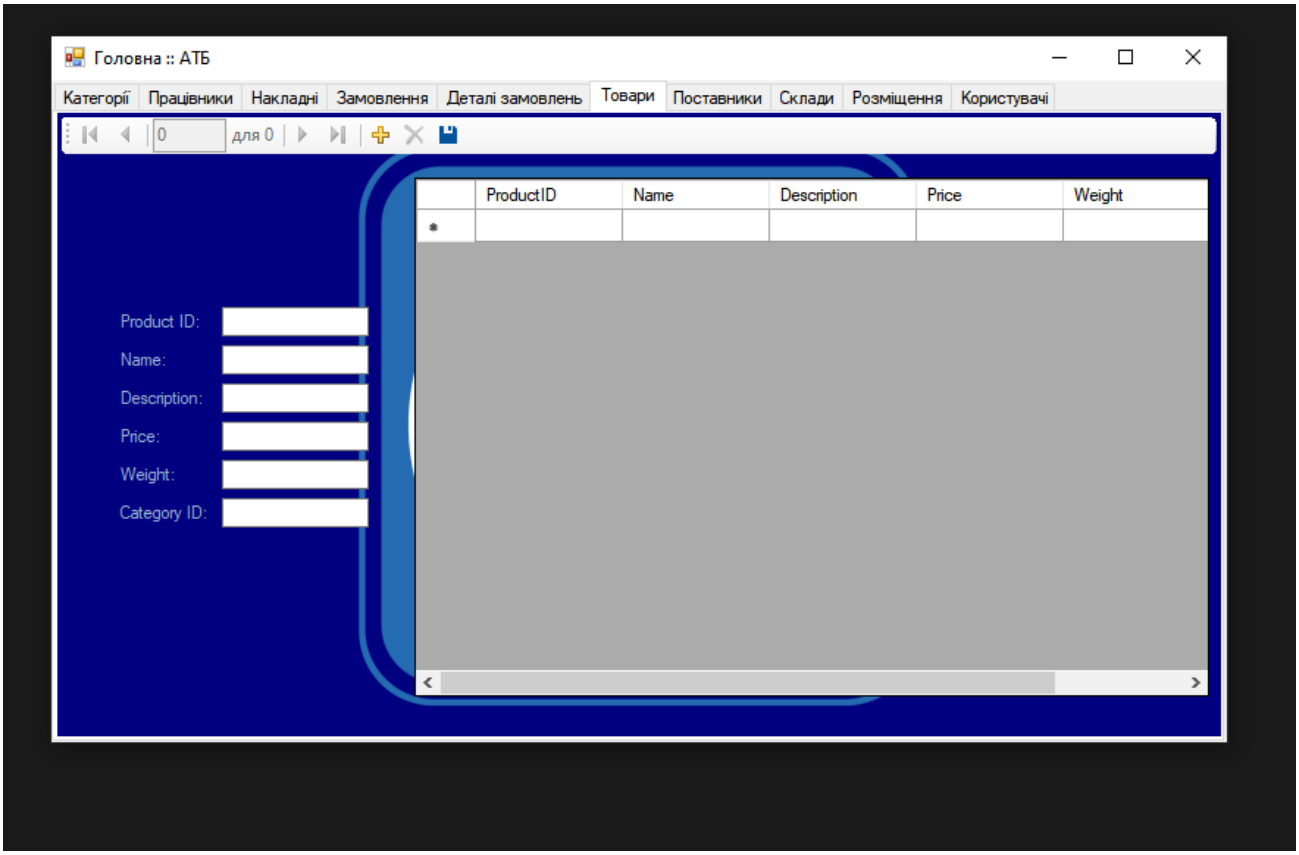

Рисунок 3.8 — Вікно товарів

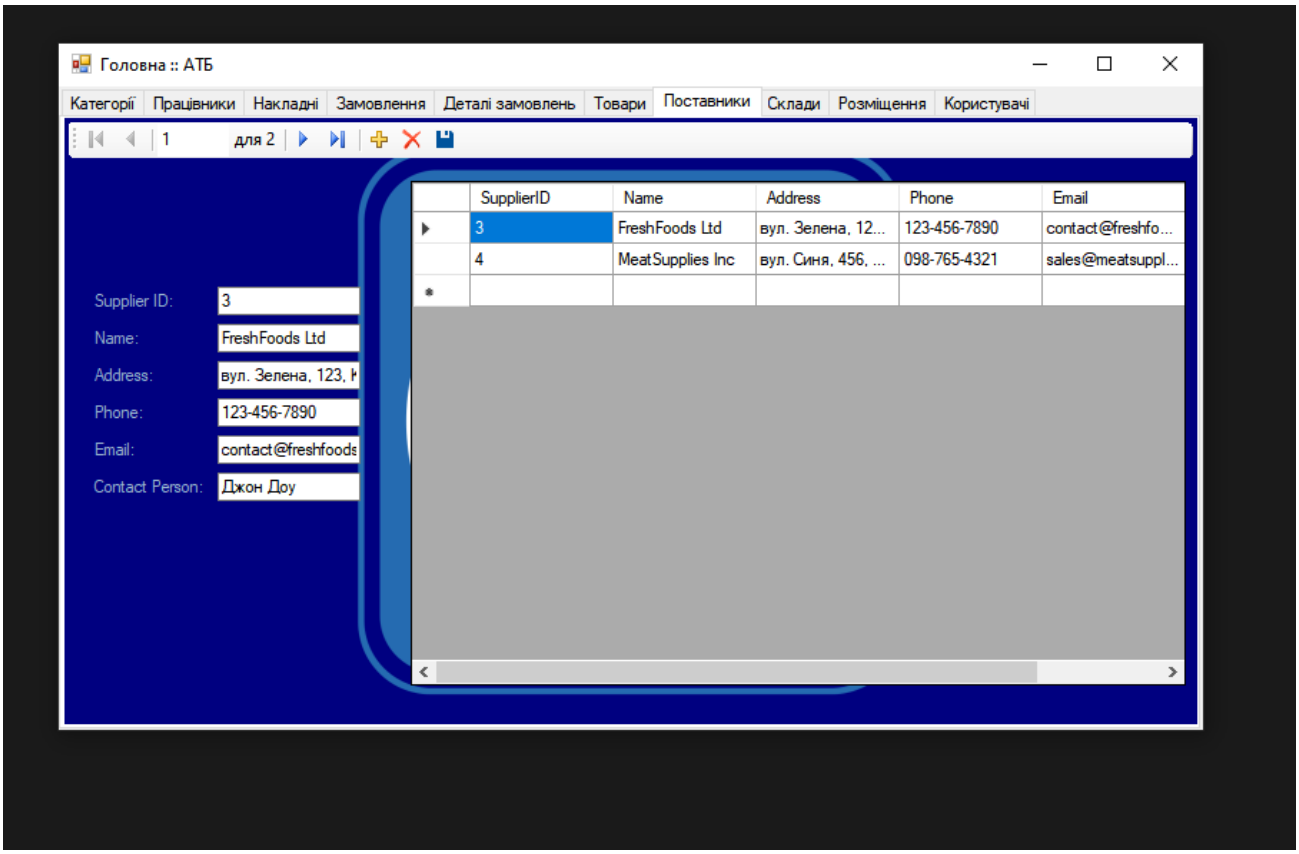

Рисунок 3.9 — Вікно поставників

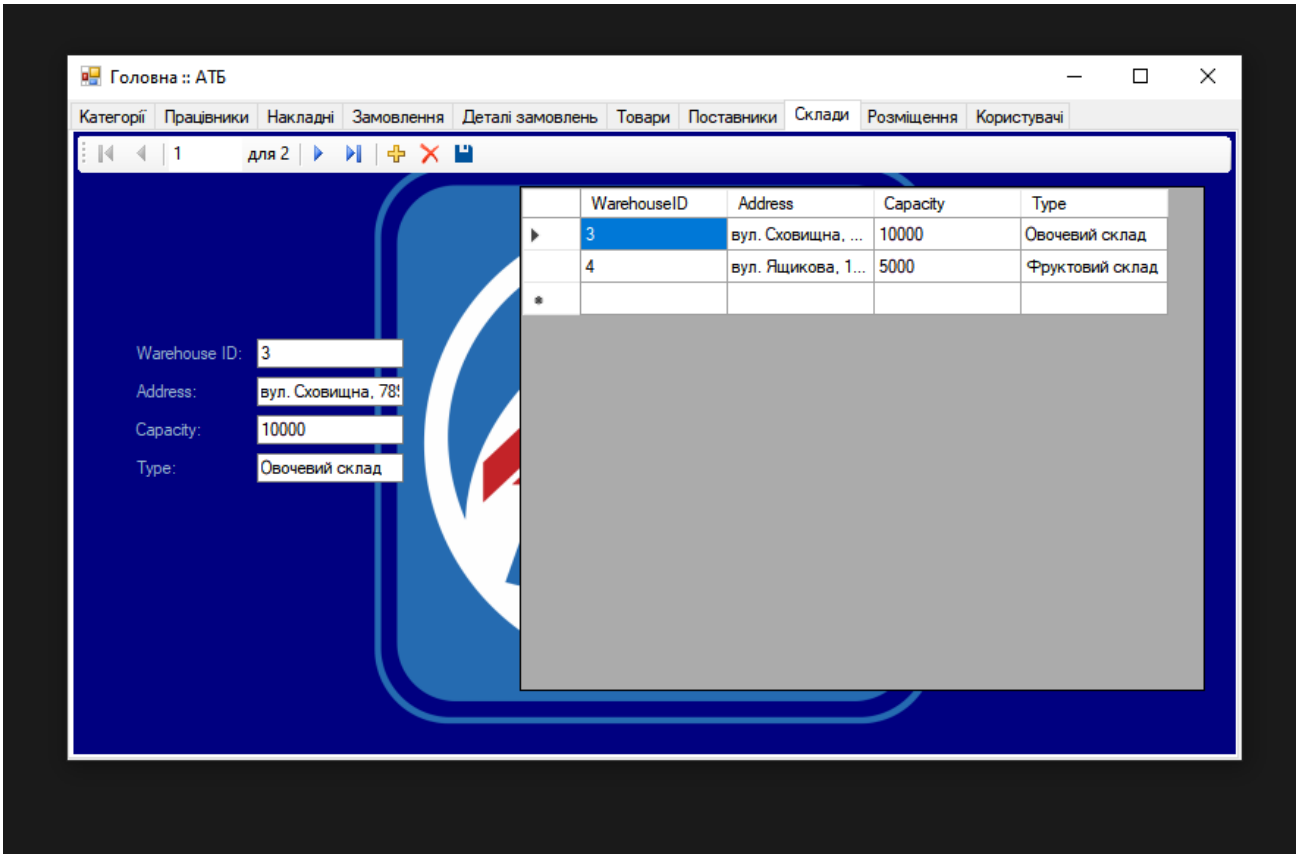

Рисунок 3.10 — Вікно складів

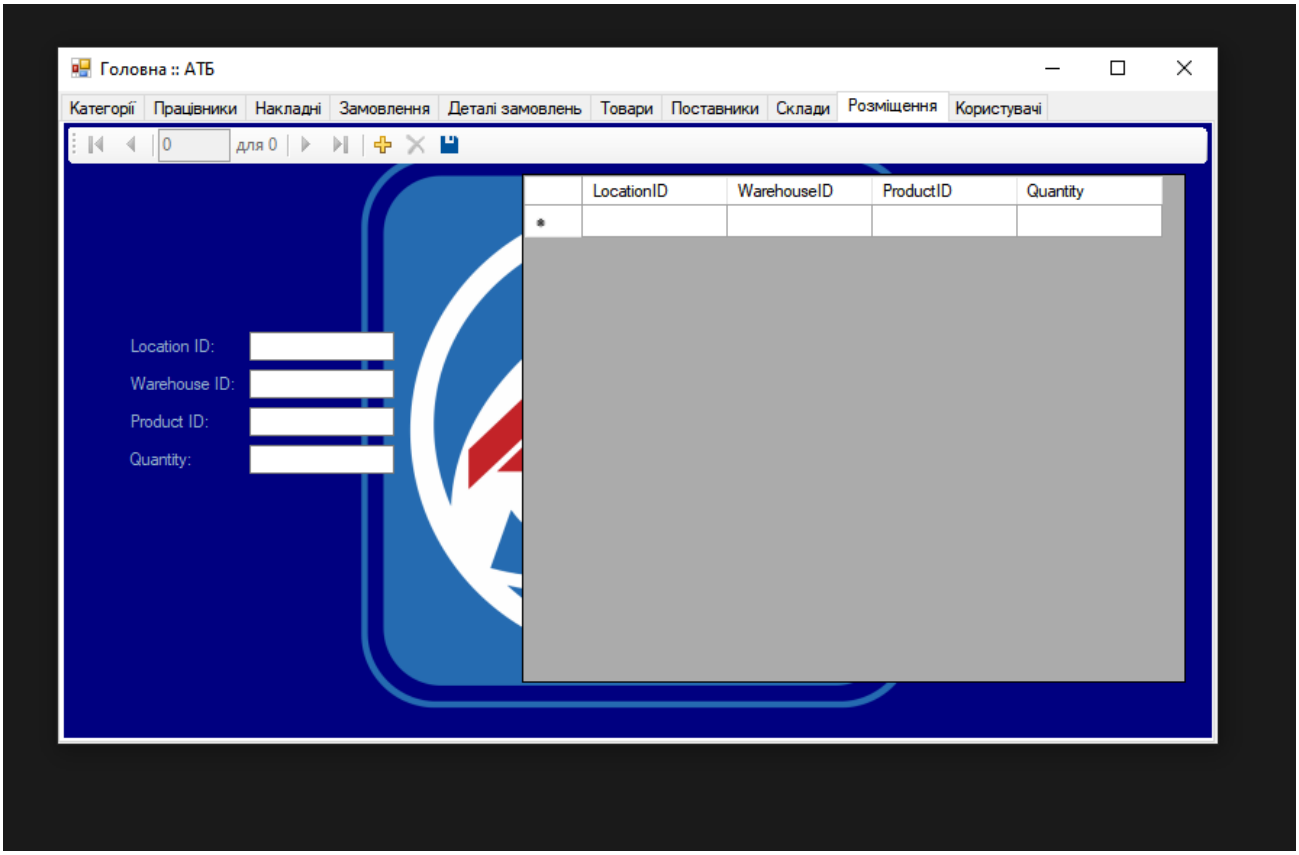

Рисунок 3.11 — Вікно розміщень

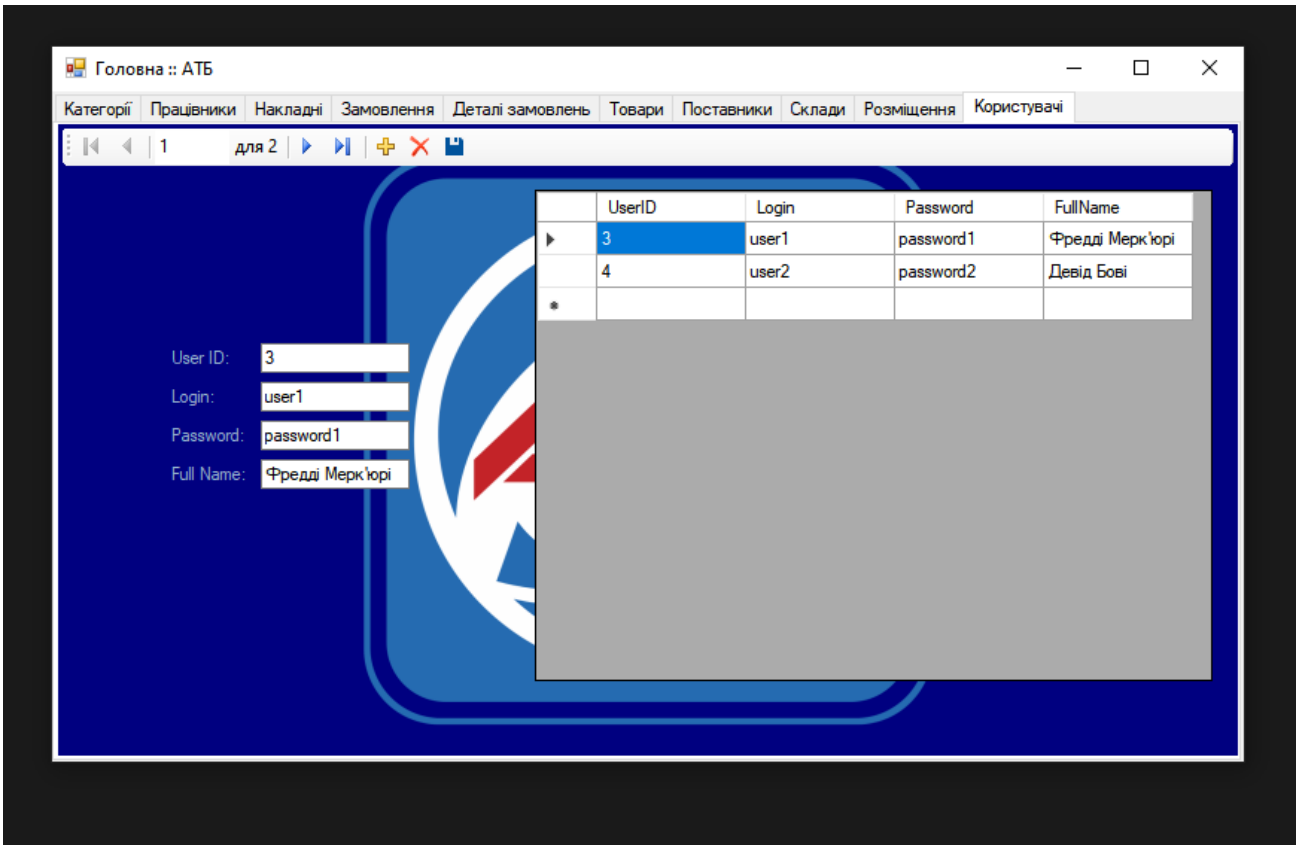

Рисунок 3.12 — Вікно користувачів

3.3 Технічне та системне забезпечення розробки

3.4.1. Обґрунтування вибору технічних засобів

№ п/п	Основні характеристики комп'ютера
Технічне забезпечення для сервера	
1	2x Intel Xeon E5-2670 v3 Cores: 2x 12x 2.30 GHz (Dual 12 Core) RAM: 192 GB DDR4 ECC reg. HDDs: 16x 4 TB SATA 7.2k RPM HW Raid IPMI/KVM
Технічне забезпечення для клієнта	
1	Intel I3-9100 Кількість ядер – 4, Кількість потоків - 4 Максимальна тактова частота у режимі Turbo - 4,20 GHz RAM: Обсяг пам'яті - 8 ГБ, Частота пам'яті - 2666 МГц; HDD: 480 ГБ ;
2	Монітор 21,5"
3	Миша USB
4	Клавіатура USB

3.4.2 Обґрунтування вибору засобів розробки

C# є мовою програмування високого рівня, призначеною для загального використання, яка підтримує багато парадигм програмування, включаючи статичну типізацію, імперативне, декларативне, функціональне, об'єктно-орієнтоване та компонентно-орієнтоване програмування. Мова була створена Андерсом Хейлсбергом з Microsoft у 2000 році і згодом стала міжнародним стандартом, схваленим Ecma (ECMA-334) і ISO/IEC (ISO/IEC 23270) у відповідно 2002 і 2003 роках. Спочатку C# була розроблена як частина .NET Framework і Visual Studio, обидва з яких були комерційними продуктами. В 2004 році був

запущений проект Mono для створення відкритого кросплатформного компілятора і середовища виконання для C#. Пізніше, Microsoft випустила відкриті продукти, такі як Visual Studio Code, Roslyn і .NET, підтримуючи C# на різних платформах. Mono пізніше було придбано Microsoft, але не було об'єднане з .NET.

На листопад 2022 року, останньою стабільною версією мови є C# 11.0, яка була представлена у рамках .NET 7.0. Під час розробки .NET Framework, бібліотеки класів спочатку були створені з використанням компілятора "Simple Managed C" (SMC). У січні 1999 року, Андерс Хейлсберг сформував команду для розробки нової мови, яку спочатку називали Cool, що стояло за "C-like Object-Oriented Language". Проте, через торговельні міркування, Microsoft вирішила перейменувати мову на C#. Коли .NET був представлений на конференції для професійних розробників у липні 2000 року, мова вже носила назву C#, а бібліотеки класів і середовище виконання ASP.NET були переписані для C#.

Андерс Хейлсберг відомий своєю ключовою роллю у розробці мови програмування C# та як провідний архітектор в Microsoft. Він має в своєму портфоліо участь у створенні таких мов, як Turbo Pascal, Embarcadero Delphi (раніше відомий під назвами CodeGear Delphi, Inprise Delphi та Borland Delphi), а також Visual J++. Хейлсберг часто висловлювався про те, що недоліки існуючих на той момент основних мов програмування, зокрема C++, Java, Delphi та Smalltalk, послужили підґрунтям для створення Common Language Runtime (CLR), що, у свою чергу, вплинув на архітектуру мови C#.

На противагу, Джеймс Гослінг, батько мови Java, та Білл Джой, співзасновник Sun Microsystems і один з розробників Java, описували C# як "імітацію" Java. Гослінг стверджував, що C# є варіантом Java, якому бракує надійності, продуктивності та безпеки. Крефт і Лангер, автори книги про потоки в C++, у своєму блозі називали Java і C# майже ідентичними мовами, критикуючи їх за відсутність інновацій. Проте, Хейлсберг у липні 2000 року заперечував, стверджуючи, що C# не є клоном Java і що її дизайн ближчий до C++, ніж до Java.

Ці дискусії та порівняння відображають змагання і розвиток у світі мов програмування, де кожна нова мова вносить вклад у загальний прогрес технологій, незалежно від взаємних впливів та спорів між їхніми творцями.

З появою C# 2.0 у листопаді 2005 року, C# і Java почали розвиватися у різних напрямках, ставши двома відмінними мовами. Однією з ключових відмінностей стало впровадження генериків у обидві мови з різними підходами до їх реалізації. C# застосовує реіфікацію, що дозволяє використовувати загальні об'єкти як повноцінні класи з динамічним генеруванням коду під час завантаження класу. Важливою особливістю C# стало також впровадження елементів функціонального програмування, зокрема LINQ, що було представлено в C# 3.0, а також підтримка лямбда-виразів, методів розширення та анонімних типів. Ці інновації надали програмістам можливості для застосування функціональних методів програмування, таких як замикання, та спростили роботу з базами даних, обробкою XML та структурами даних, зменшивши кількість шаблонного коду і зосередивши увагу на логіці програми.

Однак, символ мови C#, Енді, названий на честь Андерса Хейлсберга, був скасований у 2004 році.

C# була представлена до підкомітету ISO/IEC JTC 1 SC 22 для оцінки та затверджена згідно з міжнародними стандартами ISO/IEC 23270:2003, а пізніше з оновленнями у 2006 та 2018 роках.

В C# методи є членами класу, що можуть викликатися як функції. Сигнатура методу включає визначення доступності, тип повернення, ім'я методу та список параметрів. Ці параметри можуть мати тип, ім'я та за потреби, значення за замовчуванням. Спрощений синтаксис може бути застосований для методів, що взаємодіють з властивостями класу, але загалом оголошення класу передбачає детальне визначення методів.

У C# використання ключового слова "virtual" дозволяє підкласам замінювати методи суперкласу, подібно до механізму в C++, але відмінно від

підходу в Java, де всі неприватні методи за замовчуванням можуть бути перевизначені.

C# відзначається рядом потужних можливостей, які роблять мову гнучкою та виразною для розробників. Однією з таких можливостей є методи розширення, що дозволяють додавати нові статичні методи до існуючих класів без зміни їхнього вихідного коду, надаючи таким чином змогу розширювати функціональність об'єктів.

Використання динамічних типів у C# дозволяє методам бути пов'язаними під час виконання, схоже на механізми у JavaScript, що забезпечує гнучкість у композиції об'єктів. Це полегшує розробку динамічних застосунків, де точний тип об'єкта може бути невідомий до моменту виконання програми.

Завдяки ключовому слову "делегат", C# підтримує строго типізовані покажчики на функції, що сприяє створенню чистого та безпечного коду для подій і колбеків, схоже на механізми сигналів і слотів у Qt.

Мова також має вбудовані можливості для синхронізації викликів методів і взаємовиключних блокувань за допомогою атрибутів та ключового слова "lock", забезпечуючи безпечну роботу з ресурсами в багатопоточних застосунках.

C# відомий своїми гнучкими властивостями класів, що можуть включати методи доступу та реалізацію методів отримання та встановлення значень, полегшуючи роботу з даними, при цьому з C# 3.0 введено синтаксичний цукор для автоматично реалізованих властивостей, що спрощує оголошення властивостей.

Простори імен у C# забезпечують ізоляцію коду та схожі на пакети в Java або простори імен в C++, підтримуючи чітку організацію коду та зменшення конфліктів імен.

Нарешті, C# дозволяє використання покажчиків в небезпечних ("unsafe") блоках для прямої роботи з пам'яттю, надаючи програмістам контроль над низькорівневими операціями з пам'яттю, коли це необхідно для оптимізації або

інтеграції з некерованим кодом, водночас зберігаючи безпечну роботу з об'єктами через посилання в інших частинах програми.

У C# автоматичне збирання сміття є ключовим механізмом управління пам'яттю, що усуває необхідність в ручному звільненні пам'яті, яку об'єкти більше не використовують. Це значно знижує ризик витоків пам'яті, але вимагає уважності від розробників щодо зберігання посилань на об'єкти, щоб не створювати непотрібне навантаження на пам'ять. Коли втрачається останнє посилання на об'єкт, пам'ять, яку він займав, стає доступною для збирання сміття.

C# включає широкий спектр стандартних винятків, які розробники можуть використовувати для обробки помилок у програмах. Винятки, генеровані стандартними бібліотеками, часто мають документацію, що описує умови, за яких вони можуть бути викликані. Також, за потреби, можна створювати власні класи винятків для більш специфічних сценаріїв.

На відміну від Java, C# не використовує перевірені винятки (checked exceptions), оскільки це рішення було прийнято для спрощення розробки і зменшення складності коду, особливо у великих та розвиваючихся проектах.

Щодо успадкування, C# не підтримує множинне успадкування класів, але дозволяє класам реалізувати багато інтерфейсів. Цей підхід забезпечує гнучкість у проектуванні програм без комплікацій, які можуть супроводжувати множинне успадкування в інших мовах, таких як C++.

Коли клас реалізує декілька інтерфейсів з методами, що мають однакову сигнатуру, в C# існує можливість використати один метод для всіх інтерфейсів або, за необхідності, визначити окремі методи для кожного інтерфейсу. Це додає гнучкості в проектуванні інтерфейсів і їх реалізації в класах.

В C# перевантаження операторів дозволяє програмістам визначати поведінку стандартних операторів (наприклад, **+**, **-**, **==**, **!=** тощо) для власних класів та структур, надаючи можливість більш інтуїтивної взаємодії з об'єктами цих типів. Ця відмінність від Java, де перевантаження операторів відсутнє,

дозволяє розробникам C# створювати код, який легше читається та пишеться, особливо коли мова йде про роботу з комплексними числами, векторами та іншими математичними чи фізичними об'єктами.

LINQ (Language-Integrated Query) є могутнім інструментом у C#, що інтегрує можливості запитів безпосередньо у мову програмування, дозволяючи розробникам ефективно працювати з колекціями даних, базами даних, XML-документами та іншими джерелами даних. LINQ підтримує безпеку типів і дозволяє виконувати складні запити з перевіркою помилок на етапі компіляції, забезпечуючи високу продуктивність та зменшуючи ризик виникнення помилок у рантаймі. Мовні конструкції, такі як вирази запитів, лямбда-вирази та анонімні типи, значно спрощують роботу з даними та підвищують читабельність коду.

Типи значень у C# є важливою частиною мови, оскільки вони забезпечують ефективне зберігання та обробку примітивних даних та користувацьких структур. На відміну від посилальних типів, типи значень завжди мають значення за замовчуванням і не можуть бути **null** (окрім випадків використання **Nullable<T>** для примітивних типів). Це забезпечує вищу продуктивність для часто використовуваних даних, як-от числових значень, та дозволяє більш прямий контроль над пам'яттю. Однак, типи значень мають певні обмеження, наприклад, вони не можуть успадковуватися і не підтримують явних конструкторів за замовчуванням, що є компромісом для забезпечення їх ефективності.

Ці особливості C# відображають глибоку інтеграцію мови з .NET Framework і .NET Core, забезпечуючи розробникам широкі можливості для створення ефективних, безпечних та легко підтримуваних застосунків.

Посилальні типи в C# мають важливе значення, оскільки кожен екземпляр такого типу є унікальним, і ця унікальність визначається посиланням на конкретний об'єкт в пам'яті, а не його вмістом. Такий підхід забезпечує посилальну ідентичність, де два об'єкти вважаються рівними тільки тоді, коли вони вказують на один і той же об'єкт у пам'яті. Наприклад, тип **System.String** в

C# використовує посилальну рівність за замовчуванням, але також оптимізований для спеціального випадку, коли літерали рядків з однаковим вмістом можуть вказувати на один і той же об'єкт у пам'яті, що дозволяє ефективніше порівнювати рядки.

Система типів у C# дозволяє розробникам створювати власні типи, включаючи класи, структури, інтерфейси та перелічення, що забезпечує гнучкість у проектуванні програм та здатність до виразного кодування.

Процес стандартизації мови C# та Common Language Infrastructure (CLI) був важливим кроком у забезпеченні відкритості та доступності мови для широкого кола розробників. Спонсорство Microsoft, Hewlett-Packard і Intel у поданні специфікацій до Ecma International у 2001 році призвело до того, що C# став міжнародним стандартом. З того часу Ecma регулярно оновлює специфікації мови, включаючи нові функції, які з'явилися у мові, такі як генерики, анонімні методи та часткові класи. Ці оновлення забезпечують синхронізацію стандарту C# з розвитком мови та потребами розробників, сприяючи створенню більш надійних і ефективних програм.

Стандартизація мови C# та Common Language Infrastructure (CLI) за стандартами ISO/IEC та Ecma гарантує, що мова відповідає міжнародним вимогам і забезпечує справедливе використання патентів, що сприяє відкритості та доступності мови для широкого кола розробників. Домовленості Microsoft щодо патентів, особливо в контексті відкритого коду, демонструють зобов'язання компанії підтримувати спільноту відкритого програмного забезпечення та забезпечувати її розвиток без загрози патентних позовів у некомерційних проектах.

Зокрема, Open Specification Promise (Обіцянка відкритих специфікацій) від Microsoft є важливим кроком у напрямку забезпечення вільного доступу до технологій, що лежать в основі C#, для використання у відкритих проектах. Також угоди з Novell і захист користувачів продуктів Novell від патентних

претензій Microsoft сприяють створенню безпечного середовища для використання технологій .NET і Mono в комерційних і некомерційних проектах.

Проект Mono, який тепер є частиною Microsoft через придбання компанії Xamarin, та розробка кросплатформних інструментів з відкритим кодом, таких як Visual Studio Code і .NET Core, підкреслюють зусилля Microsoft щодо сприяння розвитку C# як відкритої, доступної та універсальної мови програмування.

Visual Studio, розроблене Microsoft, залишається одним з провідних IDE для розробки на C# та інших мовах, надаючи розробникам потужні засоби для створення, тестування та відладки програмного забезпечення. Інтеграція з .NET Framework, підтримка великої кількості мов програмування та багатий набір функціональних можливостей роблять Visual Studio незамінним інструментом для розробників різного рівня.

Visual Studio від Microsoft є одним з найпопулярніших інтегрованих середовищ розробки (IDE), що пропонує широкий спектр можливостей для розробників програмного забезпечення. Ось детальніше про його основні характеристики:

- Мови програмування: Visual Studio підтримує багато мов програмування, включаючи C#, C++, Visual Basic, F#, Python, і JavaScript, дозволяючи розробникам працювати в знайомому середовищі незалежно від того, якою мовою вони воліють користуватися.
- Інструменти для розробки: розширені можливості редактора коду з підсвічуванням синтаксису, автодоповненням, потужний відладчик, інтеграція з системами керування версіями та інші інструменти спрощують процес розробки та підвищують продуктивність.
- Візуальне проектування: завдяки вбудованим інструментам для візуального проектування, розробники можуть ефективно створювати графічні інтерфейси, проектувати бази даних і веб-сторінки, швидко налаштовуючи вигляд та поведінку своїх застосунків.

- Підтримка платформ: Visual Studio дозволяє розробляти застосунки для множини платформ, включно з Windows, Android, iOS, та веб, роблячи його ідеальним вибором для створення кросплатформного програмного забезпечення.
- Розширення та інтеграція: система розширень та плагінів дозволяє розробникам налаштувати середовище під власні потреби, інтегруючи сторонні інструменти та бібліотеки для поліпшення розробки. Visual Studio також легко інтегрується з іншими інструментами та службами, що робить його зручним у використанні в широкому спектрі розробницьких сценаріїв.

Ці характеристики роблять Visual Studio могутнім інструментом, який задовольняє потреби розробників у різноманітних областях програмування, від настільних та мобільних застосунків до складних веб-додатків і системних Visual Studio від Microsoft є потужним інструментом для розробки програмного забезпечення, який надає розробникам широкий спектр функціональності та інструментів.

Основні характеристики Visual Studio:

- Редактор коду: розширені можливості, такі як підсвічування синтаксису, автодоповнення, перехід до визначення та швидкі команди, підвищують ефективність написання коду.
- Відладчик: вбудований відладчик допомагає ефективно виявляти та виправляти помилки, дозволяючи крокувати по коду та переглядати значення змінних.
- Управління версіями: інтеграція з системами керування версіями спрощує співпрацю та відстеження змін у проектах.
- Тестування: інструменти для автоматизованого тестування дозволяють розробникам перевіряти якість коду та забезпечувати його надійність.
- Веб-розробка: вбудовані інструменти для розробки веб-додатків, включаючи підтримку веб-технологій, сприяють створенню сучасних веб-сайтів.

- Взаємодія з хмарними сервісами: інтеграція з Azure дозволяє розробникам легко створювати та керувати хмарними додатками.

Плюси Visual Studio:

- Всебічна підтримка різноманітних мов програмування та платформ.
- Велика кількість вбудованих інструментів для проектування та розробки.
- Легка інтеграція з іншими інструментами та сервісами.
- Можливість розширення функціональності за допомогою додаткових розширень.

Мінуси Visual Studio:

- Високі вимоги до системних ресурсів, що може спричинити повільну роботу на менш потужних комп'ютерах.
- Вартість платних версій може бути значною, що становить бар'єр для деяких користувачів чи компаній.
- Складність інтерфейсу для новачків, що вимагає часу для адаптації та навчання.

Попри деякі недоліки, Visual Studio залишається одним з найбільш популярних інструментів для розробки програмного забезпечення, пропонуючи широкі можливості для професіоналів у галузі.

Висновок

Visual Studio, без сумніву, є важливим інструментом у арсеналі сучасного розробника програмного забезпечення. Його універсальність та інтеграційна спроможність роблять його ідеальним середовищем для розробки широкого спектру програмних продуктів - від простих настільних застосунків до складних хмарних рішень. Попри виклики, пов'язані з високими вимогами до ресурсів та вартістю, переваги, які пропонує Visual Studio, часто переважують ці недоліки, роблячи його вибором багатьох професіоналів у галузі.

Переходячи до MS SQL Server, ми маємо справу з ще одним потужним продуктом від Microsoft, який стоїть на чолі реляційних систем управління базами даних. MS SQL Server вирізняється завдяки своїй продуктивності,

масштабованості та надійності, що робить його відмінним вибором для корпоративних застосунків, що вимагають високого рівня обробки даних.

Основні характеристики MS SQL Server

- Продуктивність і масштабованість: MS SQL Server забезпечує високу продуктивність обробки запитів та можливість масштабування для підтримки великих обсягів даних та одночасних користувачів.
- Безпека: вбудовані функції безпеки, включаючи шифрування даних, контроль доступу на основі ролей та аудит, забезпечують захист цінних даних.
- Інтеграція даних: інструменти, такі як SQL Server Integration Services (SSIS), дозволяють легко інтегрувати та перетворювати дані з різних джерел.
- Аналітика та звітність: SQL Server Analysis Services (SSAS) та Reporting Services (SSRS) пропонують потужні можливості для аналізу даних та створення звітів.

Переваги MS SQL Server:

- Надійність: доведена стабільність та надійність у вирішенні бізнес-задач.
- Широка підтримка: підтримка від Microsoft та великої спільноти користувачів забезпечує доступність ресурсів та підтримку.
- Інтеграція з іншими продуктами Microsoft: тісна інтеграція з іншими продуктами Microsoft, такими як .NET Framework, Visual Studio та Microsoft Azure, розширює можливості розробки та розгортання.

Варіанти використання MS SQL Server

MS SQL Server використовується в широкому спектрі застосунків, від корпоративних баз даних та систем управління контентом до веб-додатків та хмарних рішень. Його можливості зберігання та аналізу даних роблять його ідеальним вибором для бізнес-аналітики, CRM-систем, електронної комерції та багатьох інших сфер.

MS SQL Server продовжує бути ключовим компонентом в архітектурі багатьох рішень, завдяки своїм потужним можливостям, надійності та підтримці з боку одного з провідних розробників програмного забезпечення в світі.

РОЗДІЛ 4
ОХОРОНА ПРАЦІ

Забезпечення безпеки праці та дотримання технічних норм є ключовими елементами для системи ведення обліку товарів на логістичному хабі, маючи за мету створення безпечного середовища для робітників та запобігання інцидентам. Важливі аспекти, на які потрібно звернути увагу, включають:

1. Оцінка ризиків: важливо аналізувати потенційні ризики, що можуть виникнути під час виконання робіт у системі обліку товарів, для ідентифікації можливих загроз та розробки заходів щодо їх нейтралізації.
2. Навчання персоналу: організація кваліфікованого навчання та проведення інструктажів для працівників щодо основ безпеки, користування обладнанням, правил використання засобів індивідуального захисту та інших критичних аспектів безпеки.
3. Засоби індивідуального захисту (ЗІЗ): необхідно забезпечити доступність та коректне використання засобів індивідуального захисту, таких як захисні окуляри, рукавиці, спецодяг тощо, для зниження ризику отримання травм.
4. Технічне обслуговування та перевірка обладнання: регулярне технічне обслуговування та перевірка стану обладнання, використовуваного в системі, критично важливі для попередження збоїв та аварій.
5. Системи моніторингу та спостереження: впровадження систем моніторингу та спостереження, здатних виявляти потенційні небезпеки, відстежувати несправності та оперативно реагувати на них, включаючи системи протипожежного захисту, детектування витоків, вентиляційні системи тощо.
6. Ведення документації: комплексне документування заходів безпеки, навчальних сесій, аналізу ризиків, надзвичайних ситуацій дозволить контролювати та покращувати процеси охорони праці.

Головна стратегія полягає в попередженні ризиків, освіті персоналу, застосуванні безпечного обладнання та реалізації ефективних систем моніторингу та контролю.

ВИСНОВКИ

У ході виконання кваліфікаційної бакалаврської роботи було проведено глибоке дослідження поточних логістичних процесів мережі супермаркетів АТБ, а також розроблено та частково впроваджено інформаційну систему управління запасами.

Основні висновки роботи можна сформулювати наступним чином:

1. Автоматизація логістичних процесів є ключовим фактором для підвищення ефективності роботи мережі супермаркетів АТБ. Використання сучасних інформаційних технологій дозволяє оптимізувати процеси приймання, зберігання та відвантаження товарів, знижує ризик людських помилок та підвищує загальну продуктивність.
2. Розроблена інформаційна система забезпечує комплексний підхід до управління запасами, інтегруючи в собі всі необхідні інструменти для ефективного моніторингу та аналізу вантажопотоків. Система відповідає сучасним вимогам до безпеки праці та техніки безпеки, що дозволяє забезпечити високий рівень захисту для персоналу.
3. Економічний аналіз показав виправданість впровадження системи, оскільки очікувані вигоди від її реалізації перевищують витрати на розробку та впровадження. Термін окупності проекту, розрахований у роботі, підтверджує його ефективність та доцільність з інвестиційної точки зору.
4. Практичне впровадження системи демонструє її високу адаптивність та гнучкість, дозволяючи легко інтегрувати її в існуючі логістичні процеси мережі АТБ. Система передбачає можливість подальшого розвитку та модернізації, що робить її повністю відповідною до майбутніх потреб бізнесу.

5. Реалізація проекту вимагає комплексного підходу, який включає не тільки технічну реалізацію системи, а й навчання персоналу, адаптацію бізнес-процесів та постійний моніторинг її ефективності.

Узагальнюючи вищесказане, можна стверджувати, що розробка та впровадження інформаційної системи управління запасами дозволяє досягти висновок про досягнення усіх поставлених цілей.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. "Welcome to C# 11." 8 листопада 2022. Процитовано 26 вересня 2023.
2. "Як зробити посилання на інший рядок в C#?." Архів оригіналу за 13 грудня 2019.
3. "Архівована копія." Архів оригіналу за 7 серпня 2005. Процитовано 31 жовтня 2007.
4. "Архівована копія." Архів оригіналу за 17 травня 2008. Процитовано 31 жовтня 2007.
5. "Новые средства C# в .NET Framework 4." msdn.microsoft.com (ru-ru). 1 жовтня 2010. Архів оригіналу за 7 серпня 2019. Процитовано 7 серпня 2019.
6. "PowerShell 3 – Finally on the DLR!." web.archive.org (en-us). 28 квітня 2012. Архів оригіналу за 28 квітня 2012. Процитовано 7 серпня 2019.
7. "Deploying an Interop Application." docs.microsoft.com (en-us). Архів оригіналу за 7 серпня 2019. Процитовано 7 серпня 2019.
8. "C# Preprocessor Directives." Архів оригіналу за 4 липня 2008. Процитовано 27 лютого 2009.
9. "Standard ECMA-334 C# Language Specification, 4th edition (June 2006)." Архів оригіналу за 31 жовтня 2010. Процитовано 31 жовтня 2007.
10. "ISO/IEC 23270:2003, Information technology — C# Language Specification." Архів оригіналу за 30 червня 2006. Процитовано 31 жовтня 2007.
11. "Interoperability Principles." Архів оригіналу за 22 червня 2013. Процитовано 4 травня 2010.
12. "Patent Pledge for Open Source Developers." Архів оригіналу за 22 червня 2013. Процитовано 4 травня 2010.
13. "Patent Cooperation Agreement - Microsoft & Novell Interoperability Collaboration." Microsoft. 2 листопада 2006. Архів оригіналу за 22 червня 2013. Процитовано 5 липня 2009.

14. "Definitions." Microsoft. 2 листопада 2006. Архів оригіналу за 22 червня 2013. Процитовано 5 липня 2009.
15. Steinman, Justin (7 листопада 2006). "Novell Answers Questions from the Community." Архів оригіналу за 22 червня 2013. Процитовано 5 липня 2009.
16. "Covenant to Downstream Recipients of Moonlight — Microsoft & Novell Interoperability Collaboration." Microsoft. 28 вересня 2007. Архів оригіналу за 2 березня 2012. Процитовано 8 березня 2008.
17. Stallman, Richard (26 червня 2009). "Why free software shouldn't depend on Mono or C#." Free Software Foundation. Архів оригіналу за 22 червня 2013. Процитовано 2 липня 2009.
18. "Microsoft's Empty Promise." Free Software Foundation. 16 липня 2009. Архів оригіналу за 22 червня 2013. Процитовано 2009-078-03. «Until that happens, free software developers still should not write software that depends on Mono. C# implementations can still be attacked by Microsoft's patents: the Community Promise is designed to give the company several outs if it wants them. We don't want to see developers' hard work lost to the community if we lose the ability to use Mono, and until we eliminate software patents altogether, using another language is the best way to prevent that from happening.»
19. The ECMA C# and CLI Standards. 6 липня 2009. Архів оригіналу за 22 червня 2013. Процитовано 2009-078-03.
20. Gubskiy, Andrew (4 квітня 2023). The C# Multiverse — the Singularity of Programming Languages. Medium (англ.). Процитовано 22 червня 2023.

ДОДАТОК А
SQL-ЗАПИТИ

```
-- 1. Категорії товарів
CREATE TABLE Categories (
    CategoryID INT PRIMARY KEY IDENTITY,
    Name NVARCHAR(255),
    Description NVARCHAR(MAX)
);

-- 2. Постачальники
CREATE TABLE Suppliers (
    SupplierID INT PRIMARY KEY IDENTITY,
    Name NVARCHAR(255),
    Address NVARCHAR(MAX),
    Phone NVARCHAR(50),
    Email NVARCHAR(255),
    ContactPerson NVARCHAR(255)
);

-- 3. Співробітники
CREATE TABLE Employees (
    EmployeeID INT PRIMARY KEY IDENTITY,
    Name NVARCHAR(255),
    Position NVARCHAR(255),
    Phone NVARCHAR(50),
    Email NVARCHAR(255),
    Address NVARCHAR(MAX)
);
```

```
-- 4. Склад
CREATE TABLE Warehouse (
    WarehouseID INT PRIMARY KEY IDENTITY,
    Address NVARCHAR(MAX),
    Capacity INT,
    Type NVARCHAR(255)
);

-- 5. Товари
CREATE TABLE Products (
    ProductID INT PRIMARY KEY IDENTITY,
    Name NVARCHAR(255),
    Description NVARCHAR(MAX),
    Price DECIMAL(10, 2),
    Weight DECIMAL(10, 2),
    CategoryID INT FOREIGN KEY REFERENCES Categories(CategoryID)
);

-- 6. Замовлення від постачальників
CREATE TABLE Orders (
    OrderID INT PRIMARY KEY IDENTITY,
    SupplierID INT FOREIGN KEY REFERENCES Suppliers(SupplierID),
    OrderDate DATE,
    Status NVARCHAR(50),
    ExpectedDeliveryDate DATE
);

-- 7. Деталі замовлень
```

```
CREATE TABLE OrderDetails (
    OrderDetailID INT PRIMARY KEY IDENTITY,
    OrderID INT FOREIGN KEY REFERENCES Orders(OrderID),
    ProductID INT FOREIGN KEY REFERENCES Products(ProductID),
    Quantity INT,
    UnitPrice DECIMAL(10, 2)
);

-- 8. Прийом товарів
CREATE TABLE GoodsReceipt (
    ReceiptID INT PRIMARY KEY IDENTITY,
    OrderID INT FOREIGN KEY REFERENCES Orders(OrderID),
    ReceiptDate DATE,
    ReceivedBy INT FOREIGN KEY REFERENCES Employees(EmployeeID)
);

-- 9. Розташування товарів на складі
CREATE TABLE WarehouseLocations (
    LocationID INT PRIMARY KEY IDENTITY,
    WarehouseID INT FOREIGN KEY REFERENCES Warehouse(WarehouseID),
    ProductID INT FOREIGN KEY REFERENCES Products(ProductID),
    Quantity INT
);

-- 10. Користувачі
CREATE TABLE Users (
    UserID INT PRIMARY KEY IDENTITY,
    Login NVARCHAR(255),
```

```
    Password NVARCHAR(255),
    FullName NVARCHAR(255)
);
```

ДОДАТОК Б
ЛІСТИНГ ПРОГРАМНОГО КОДУ

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace ATBStoraging
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
```

```
            string connectionString = "Data Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\\Storaging DB.mdf;Integrated Security=True";
            string query = "SELECT COUNT(1) FROM Users WHERE Login=@login AND Password=@password";

            using (SqlConnection conn = new SqlConnection(connectionString))
            {
                conn.Open();
                using (SqlCommand cmd = new SqlCommand(query, conn))
                {
                    cmd.Parameters.AddWithValue("@login", textBox1.Text.Trim());
                    cmd.Parameters.AddWithValue("@password", textBox2.Text.Trim());

                    int result = Convert.ToInt32(cmd.ExecuteScalar());
                    if (result == 1)
                    {
                        MessageBox.Show("Успішний вхід!");
                        this.Hide();
                        MainForm mainForm = new MainForm();
                        mainForm.ShowDialog();
                        this.Show();

                    }
                    else
                    {
                        MessageBox.Show("Логін або пароль невірний.");
                    }
```

```
            }
        }
    }
}

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace ATBStoraging
{
    public partial class MainForm : Form
    {
        public MainForm()
        {
            InitializeComponent();
        }

        private void categoriesBindingNavigatorSaveItem_Click(object sender,
EventArgs e)
        {
            this.Validate();
```

```
            this.categoriesBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.storagingDBDataSet);

        }

        private void categoriesBindingNavigatorSaveItem_Click_1(object sender,
EventArgs e)
        {
            this.Validate();
            this.categoriesBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.storagingDBDataSet);

        }

        private void categoriesBindingNavigatorSaveItem_Click_2(object sender,
EventArgs e)
        {
            this.Validate();
            this.categoriesBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.storagingDBDataSet);

        }

        private void categoriesBindingNavigatorSaveItem_Click_3(object sender,
EventArgs e)
        {
            this.Validate();
            this.categoriesBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.storagingDBDataSet);
```

```
        }

        private void categoriesBindingNavigatorSaveItem_Click_4(object sender,
EventArgs e)
        {
            this.Validate();
            this.categoriesBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.storagingDBDataSet);

        }

        private void categoriesBindingNavigatorSaveItem_Click_5(object sender,
EventArgs e)
        {
            this.Validate();
            this.categoriesBindingSource.EndEdit();
            this.tableAdapterManager.UpdateAll(this.storagingDBDataSet);

        }

        private void MainForm_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу
"storagingDBDataSet.Users". При необходимости она может быть перемещена или
удалена.
            this.usersTableAdapter.Fill(this.storagingDBDataSet.Users);
```

// TODO: данная строка кода позволяет загрузить данные в таблицу "storagingDBDataSet.WarehouseLocations". При необходимости она может быть перемещена или удалена.

this.warehouseLocationsTableAdapter.Fill(this.storagingDBDataSet.WarehouseLocations);

// TODO: данная строка кода позволяет загрузить данные в таблицу "storagingDBDataSet.Warehouse". При необходимости она может быть перемещена или удалена.

this.warehouseTableAdapter.Fill(this.storagingDBDataSet.Warehouse);

// TODO: данная строка кода позволяет загрузить данные в таблицу "storagingDBDataSet.Suppliers". При необходимости она может быть перемещена или удалена.

this.suppliersTableAdapter.Fill(this.storagingDBDataSet.Suppliers);

// TODO: данная строка кода позволяет загрузить данные в таблицу "storagingDBDataSet.Products". При необходимости она может быть перемещена или удалена.

this.productsTableAdapter.Fill(this.storagingDBDataSet.Products);

// TODO: данная строка кода позволяет загрузить данные в таблицу "storagingDBDataSet.OrderDetails". При необходимости она может быть перемещена или удалена.

this.orderDetailsTableAdapter.Fill(this.storagingDBDataSet.OrderDetails);

// TODO: данная строка кода позволяет загрузить данные в таблицу "storagingDBDataSet.Orders". При необходимости она может быть перемещена или удалена.

this.ordersTableAdapter.Fill(this.storagingDBDataSet.Orders);

// TODO: данная строка кода позволяет загрузить данные в таблицу "storagingDBDataSet.GoodsReceipt". При необходимости она может быть перемещена или удалена.

this.goodsReceiptTableAdapter.Fill(this.storagingDBDataSet.GoodsReceipt);

// TODO: данная строка кода позволяет загрузить данные в таблицу "storagingDBDataSet.Employees". При необходимости она может быть перемещена или удалена.

this.employeesTableAdapter.Fill(this.storagingDBDataSet.Employees);

// TODO: данная строка кода позволяет загрузить данные в таблицу "storagingDBDataSet.Categories". При необходимости она может быть перемещена или удалена.

this.categoriesTableAdapter.Fill(this.storagingDBDataSet.Categories);

}

}

}